UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 3

COMPONENTE INTERNACIONALIZACIÓN PARA LOS SISTEMAS QUE SE CONSTRUYEN SOBRE EL MARCO DE TRABAJO SAUXE

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autores:

Odalys Carballosa Fernández

Yunior Ismael Charro Pérez

Tutores:

Ing. Claudia Bravo Batista

Ing. René R. Bauta Camejo

La Habana, junio de 2015

"Año 57 de la Revolución"

"No busques ser alguien de éxito, sino busca ser alguien valioso:
lo demás llegará naturalmente."
Albert Einstein"

DECLARACIÓN DE AUTORÍA

Declaramos ser los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

Odalys Carballosa Fernández

_____
Firma del Autor

Yunior Ismael Charro Pérez

_____
Firma del Autor

Claudia Bravo Batista

_____
Firma del Tutor

René Rodrigo Bauta Camejo

_____
Firma del Tutor

DATOS DE CONTACTO

Síntesis de los tutores:

Tutor: Ing. René R. Bauta Camejo

Graduado de Ingeniería en Ciencias Informáticas en julio del 2009

Dirección: Universidad de las Ciencias Informáticas (UCI), Edificio: 129, Apto: 105

Teléfono Oficina: +53 – 7 – 8358296. Teléfono Apto: +53 – 7 – 837 – 3156

E-mail: rrbauta@uci.cu

Categoría docente: Instructor

Tutor: Ing. Claudia Bravo Batista

Graduado de Ingeniería en Ciencias Informáticas en julio del 2012

Dirección: Universidad de las Ciencias Informáticas (UCI), Edificio: 27, Apto: 201

Teléfono Oficina: +53 – 7 – 8358292. Teléfono Apto: +53 – 7 – 835 – 8767

E-mail: cbravo@uci.cu

DEDICATORIA

A mis padres y mi hermano por su sacrificio, entrega y amor incondicional.

A mi abuela por apoyarme y brindarme su amor sin límites.

Odalys

A mis padres por ser mi guía, a mi hermano por ser mi ejemplo a seguir, a mi novia Thais por estar siempre presente y quererme tal y como soy, y a mi hijo que está por nacer.

Yunior

AGRADECIMIENTOS

De Odalys

A mi mamá por ser la persona más importante que tengo en la vida, por darme siempre su amor sin límites, por confiar en mí en todo momento, por su sacrificio incondicional, por ser mi guía, mi ejemplo y por darme fuerzas para seguir adelante, te amo mamita. A mi papá que a pesar de su enfermedad sé que me ama y que soy su niña linda. A mi hermano por ser mi amigo, aconsejarme y hacer función de padre cuando lo necesité, Oss te quiero con la vida. A mamina por ser mi abuela bella, y siempre apoyarme en todo. A Lary por ser mi segunda madre, mi hermana, mi amiga y por estar ahí siempre. A mi bebsote por soportar mi intensidad y por estar ahí dándome fuerzas para seguir adelante cuando pensé que el mundo se me caía encima, te amo Jordy. A Yailín por su amistad, por ser la mejor cuñada y por estar ahí siempre. A tía, a tata, a tío pite, a todos mis primos, a toda mi hermosa familia. A Dayo mi amiga de toda la vida. A la yabosita por sus consejos y más que suegra ser mi amiga. A mi tutora por su ayuda sin límites, por su amistad, por estar disponible a la hora que fuese, Clau gracias por todo, sin ti esto hubiese sido muy difícil. A mis amigos del proyecto, al papi happy por alegrarnos el día con sus ocurrencias y broncas con Clau, a Katia por estar ahí para ayudarnos en todo, al Ino y al bestial. A mi familia del 3503 el mejor grupo de la UCI, nunca los voy a olvidar. A Roly, Burgui, el Yaicel, Ernesto, los jimi, Bauti, Sosa, Jose y a mi compañero de tesis Yunior. A Ale por ser mi consejero y amigo. A Negrín por siempre sacarme una sonrisa en los momentos difíciles, en esos momentos de mucho estrés. A mi hermano de la UCI, Ani gracias por ser mi paño de lágrimas, mi compañero y amigo. A Marian por su amistad y sinceridad en todo momento. A Araí por los buenos momentos que pasamos en estos años, a Aliss por esas madrugadas arreglando el documento. A esas profes que llegaron para quedarse, Maigre mi amiga de todas las batallas, Daimara y Lizandra por estar ahí siempre. A Yisel por brindarnos su ayuda, a los profes del tribunal por sus sabios consejos. A todas las personas que he conocido en la UCI y me han brindado una mano amiga. A todos los que de una forma u otra aportaron su granito de arena para que hoy yo fuera una profesional. Muchas gracias.

AGRADECIMIENTOS

De Yunior

A toda a mi familia especialmente a mis padres que siempre han sido mi guía, a mi hermano que es y será siempre mi ejemplo a seguir, a mi novia Thais que ha estado a mi lado en las buenas y en las malas, a mi tía, mi tío y mis primos de Santiago que también son padres y hermanos para mí, a mis tíos y primos de Las Tunas que aunque no los veo muy seguido siempre los llevo en el corazón.

A mi súper tutora Claudia por su dedicación, paciencia y por ser una más del equipo.

A mi segunda familia, la que formé aquí en la UCI, Rolando, Negrín, Bauta, Yaicel, Marielys, Ernesto, los jimas, Jose Angel, Burgos, Alicia, a mi compañera de tesis Odalys, a todos los profesores que me impartieron clases y que de todos aprendí algo y a todos los que alguna vez fueron parte de mi vida en esta escuela.

RESUMEN

Los procesos de construcción de software deben adaptarse con mayor rapidez a un mundo cada vez más globalizado. Es por ello que se internacionalizan las aplicaciones web. De esta forma se puede adaptar el software a distintos idiomas, brindando la posibilidad a usuarios con diferentes ámbitos geográficos tener acceso al sistema sin ningún inconveniente. Para realizar la configuración del idioma y las regiones del marco de trabajo Sauxe, desarrollado en el departamento de Desarrollo de Componente, se necesita de un tiempo considerable. Para ello se tiene en cuenta que las configuraciones de los ficheros del idioma de cada componente se realizan de forma manual por parte de los especialistas. El presente trabajo está orientado a la creación de un componente que reduzca el tiempo de configuración del idioma y las regiones de las aplicaciones que se construyen sobre el marco de trabajo Sauxe. Para el desarrollo de la solución se realizó un análisis de los conceptos asociados al tema, así como de varias soluciones existentes que aportaron aspectos de importancia. En el trabajo se describe la solución propuesta y se realiza un análisis de los resultados obtenidos durante la validación.

Palabras claves: componente, idioma, internacionalización, regionalización

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

En la actualidad el acceso a las tecnologías de la información y las comunicaciones se limita esencialmente por la adaptación del contenido a diferentes idiomas y regiones, proceso que se define como internacionalización en el libro Diccionario Ilustrado Océano de la Lengua Española. La internacionalización de un sistema informático permite además usar distintos tipos de formatos de fecha, y la localización por su parte selecciona el adecuado para una región.

En Cuba, la Universidad de las Ciencias Informáticas (UCI) ha creado diferentes centros de producción, entre los que se encuentra el Centro de Informatización de Entidades (CEIGE). El mismo encamina sus esfuerzos al desarrollo de sistemas de gestión para el entorno empresarial mediante el uso de tecnologías web. En dicha área radica el departamento de Desarrollo de Componentes donde se construye el marco de trabajo Sauxe, teniendo como objetivos agilizar el proceso de desarrollo, brindar una base tecnológica en el desarrollo de aplicaciones web para la gestión de entidades y el apoyo a la toma de decisiones.

La configuración de los idiomas inglés y español para todos los componentes del marco de trabajo, fue realizada por dos especialistas del departamento, el tiempo dedicado por parte de los profesionales para la realización de la configuración fue de tres meses, durante los meses de enero, febrero y marzo del año 2014, quedando registradas las actualizaciones del código fuente en el control de versiones, trabajando ocho horas diarias en dependencia de la complejidad del componente, con una disponibilidad de dos puestos de trabajo y haciendo uso de traductores para la definición de los términos. Partiendo de la información descrita con anterioridad, obtenida por medio de las entrevistas abiertas realizadas a estos especialistas en ese mismo año, se propuso ejecutar una encuesta enfocada en el tiempo y esfuerzo que requería realizar la configuración de los idiomas.

La población a analizar estuvo constituida por los profesionales del departamento de Desarrollo de Componentes para un total de 12 ingenieros. La muestra se seleccionó de forma intencionada teniendo en cuenta que todos los elementos de la población tienen características similares, en este caso, especialistas que han configurado el idioma en el marco de trabajo Sauxe, además se seleccionaron los profesionales, en los cuales se pudieran obtener los datos necesarios en el período antes de aplicar la propuesta, de estos se escogieron cincos, que representa el 41% de la población.

Los resultados de la encuesta expusieron que todos los especialistas habían realizado la configuración del idioma español y solamente dos del idioma inglés, y que se requería de un esfuerzo considerable para la ejecución de la tarea, porque se hace necesario insertar y configurar los ficheros del idioma con extensión

.json, en los cuales se definen las etiquetas que se encuentran en el código fuente javascript y php. Además, se debe importar el fichero del idioma con extensión .js para los componentes definidos por la librería ExtJS y se precisa del uso de traductores para la definición de los términos, en el caso de la configuración del idioma inglés.

En general, los especialistas disponían entre dos y tres horas de trabajo, para realizar la configuración de un idioma por cada componente desarrollado en el marco de trabajo Sauxe, teniendo en cuenta la complejidad del mismo, y para realizar la localización y la configuración de los formatos de la fecha y la hora de 15 minutos, en dependencia de las modificaciones efectuadas en el código fuente de la capa de presentación.

Teniendo en cuenta lo anteriormente expuesto, se identifica como **problema a resolver**: ¿Cómo reducir el tiempo de configuración del idioma y las regiones de las aplicaciones que se construyen sobre el marco de trabajo Sauxe?

Queda definido como **objeto de estudio:** el proceso de internacionalización en las aplicaciones web y se enmarca la investigación en el **campo de acción:** configuración de los idiomas que se definen en el marco de trabajo ExtJS en su versión 4.2.1, la zona horaria, la fecha, la hora y el calendario en el marco de trabajo Sauxe.

Con el fin de darle respuesta al problema planteado se identifica el siguiente **objetivo general:** desarrollar un componente de internacionalización que permita disminuir el tiempo de configuración del idioma y las regiones de las aplicaciones que se construyen sobre el marco de trabajo Sauxe.

Objetivos específicos

1. Confeccionar el marco teórico de la investigación sobre la internacionalización en aplicaciones web para definir una propuesta de solución.
2. Indentificar los requisitos funcionales y no funcionales del componente Internacionalización para el marco de trabajo Sauxe.
3. Realizar el análisis y diseño del componente Internacionalización para el marco de trabajo Sauxe.
4. Realizar la implementación del componente Internacionalización para el marco de trabajo Sauxe.
5. Validar la implementación de la solución mediante pruebas de Caja Blanca y Caja Negra.
6. Validar la investigación utilizando el método pre-exprimento.

Idea a defender

Si se desarrolla un componente de internacionalización se logrará disminuir el tiempo de configuración del idioma y las regiones de las aplicaciones que se construyen sobre el marco de trabajo Sauxe.

Resultados esperados

Se desea alcanzar con el desarrollo de la investigación, la construcción de un componente que permita internacionalizar los sistemas que se construyen sobre el marco de trabajo Sauxe.

Métodos científicos a utilizar

Métodos Teóricos

Histórico – Lógico: se aplica con el objetivo de conocer la existencia de aplicaciones web que realicen el proceso de internacionalización, para observar la ejecución del proceso en estos tipos de sistemas y utilizarlos como puntos de referencia**.**

Analítico – Sintético: se aplica en el estudio de la bibliografía concretando aquella de interés para la investigación con el propósito de determinar los marcos de trabajo que representaban soluciones aplicables a la problemática planteada.

Modelación: se aplica para generar los artefactos que se construyen durante el proceso de Ingeniería de Software, como modelo para comprender las propiedades, funcionalidades y relaciones que tiene el componente Internacionalización.

Métodos Empíricos

Entrevista: se utilizó con la finalidad de buscar información sobre la situación existente. Permitió reunir un conjunto de características necesarias para poder dimensionar el problema de la investigación. Se realizaron entrevistas abiertas a los dos especialistas que configuraron el idioma en el marco de trabajo Sauxe, donde se efectuaron preguntas enfocadas en describir la configuración de los temas de internacionalización ver Anexo 1.

Encuesta: se utilizó con la finalidad de conocer sobre los motivos principales para realizar un componente de internacionalización y su importancia. La misma fue realizada utilizando la metodología que plantea Leonor Buendía (Buendía, 1998), la cual establece tres fases de desarrollo: teórico conceptual, metodológica y estadístico - conceptual; en la primera fase incluye el planteamiento de los objetivos y/o problemas e hipótesis de investigación, en el segundo la selección de la muestra y la definición de las variables que van a ser objeto de estudio y en la tercera se incluye la elaboración piloto y definitiva del cuestionario y la codificación del mismo que permitirá establecer las conclusiones correspondientes al

estudio (Figura.1). En el Anexo 2 se muestra la encuesta y el análisis realizado a partir de la metodología propuesta por Buendía (1998).

*Figura 1.Fases de desarrollo de una encuesta según Buendía (1998***)**

Para dar cumplimiento a los objetivos trazados, el presente documento estará estructurado en tres capítulos. A continuación se expone una breve descripción de cada uno de ellos.

Capítulo I: Fundamentación Teórica

En este capítulo se establece una relación entre los conceptos fundamentales en aras de precisar aquellos que constituirán la base teórica de la presente investigación. Se efectúa un estudio del estado del arte referente a la internacionalización en las aplicaciones web. Además se hace una descripción de las tecnologías que se emplearán, de la metodología definida y aprobada para el desarrollo de proyectos de la UCI, que es una variación de la metodología: Proceso Unificado Ágil (Agile Unified Process por sus siglas en inglés (AUP)) en unión con el modelo CMMI-v1.3, que guiará todo el proceso de la construcción de la herramienta.

Capítulo II: Propuesta de solución

En este capítulo se generan los artefactos por cada una de las disciplinas del proceso de desarrollo de software, propuestos por la Metodología de desarrollo para la Actividad productiva de la UCI. En la disciplina de Modelo de negocio se realiza el Modelo conceptual. En la disciplina de Requisitos se describen las técnicas de captura de requisitos establecidas y aplicadas, se realiza la administración y validación de los requisitos y se aplican técnicas de agrupación de requisitos. En la disciplina de Análisis y

diseño se define la arquitectura donde se identifican los patrones y estilos arquitectónicos utilizados y se modela el sistema para que soporte todos los requisitos.

Capítulo III: Implementación y pruebas

En este capítulo se analizan los estándares de codificación utilizados para lograr un mejor entendimiento y legibilidad del código. Se exponen los resultados de la aplicación de las métricas Tamaño Operacional de Clases y Relaciones entre Clases al diseño elaborado. Se generan los artefactos de la disciplina de Implementación, a través de la construcción del Diagrama de componentes y de la disciplina de Pruebas realizando pruebas internas y pruebas de aceptación. En las pruebas internas se muestran y describen los resultados de las pruebas de Caja Blanca aplicando el método del Camino básico y Caja Negra aplicando el método de Partición equivalente realizadas a la solución para demostrar su correcto funcionamiento. Se realiza un diseño pre-experimental para validar la investigación utilizando como instrumento para medir la variable operacional las encuestas.

CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA

Introducción

La internacionalización en un sistema computacional que permite adaptar el software a diferentes idiomas y regiones. En el presente capítulo se realiza el marco conceptual sobre los principales temas relacionados con la investigación. Se analizan diversos marcos de trabajo que efectúan la internacionalización, para observar la ejecución del proceso en estos tipos de sistemas. Además se identifica la metodología que guiará el proceso de desarrollo del componente, así como las herramientas que se utilizarán para la construcción del mismo.

1.1 Conceptos asociados al dominio del problema

Componente: un componente es un elemento de software que se ajusta a un modelo de componentes y que puede ser desplegado y compuesto de forma independiente sin modificación según un estándar de composición (Councill, 2001).

Ian Sommerville (Sommerville, 2005) afirma que los componentes deben ser:

- ✓ **Estandarizados**: significa que estos deben ajustarse a algún modelo de componentes.
- ✓ **Independientes**: deben poder componerse y desplegarse sin tener que utilizar otros componentes.
- ✓ **Componibles**: las interacciones externas deben ser a través de interfaces definidas públicamente.
- ✓ **Desplegables**: deben ser capaces de funcionar como una entidad autónoma o sobre una plataforma de componentes que implemente el modelo de componentes.
- ✓ **Documentados**: tienen que estar completamente documentados para que los usuarios potenciales puedan decidir si los componentes satisfacen sus necesidades.

El análisis de este concepto permite tener un dominio de lo que se desea desarrollar para darle solución al problema a resolver, en este caso, se debe construir un elemento de software que debe ajustarse a un modelo de componentes. También debe componerse y desplegarse de forma independiente sobre una plataforma que implemente un modelo de componentes.

Internacionalización: según Lawrence Welch y Reijo Luostarinen la internacionalización se entiende por todo aquel conjunto de operaciones que faciliten el establecimiento de vínculos más o menos estables entre las empresas y los mercados internacionales, a lo largo de un proceso de creciente implicación y proyección internacional (Welch, y otros, 1988).

Según el Diccionario Ilustrado Océano de la Lengua Española, "*la internacionalización es el diseño y desarrollo de un producto, una aplicación o el contenido de un documento de modo tal que permita una fácil localización (adaptación del contenido a diferentes idiomas, regiones o culturas) sin necesidad de realizar cambios en el código"*. (Diccionario Ilustrado Océano de la Lengua Española. Edición del Milenio)

La mayoría de los autores coinciden que la internacionalización es el proceso que permite que un programa ofrezca a un usuario un entorno de computación adaptado a su propio país: lenguaje, formatos de fecha y hora. Para la investigación se utilizará la siguiente definición ya que abarca varios de los elementos comunes de las otras definiciones: *"La internacionalización a menudo abreviada como i18n, es el proceso de generalizar un producto para que pueda manejar múltiples idiomas y convenciones culturales sin la necesidad de rediseñarlo"* (Moreno, 2011).

Localización: "*es el proceso de adaptar el software para una región específica mediante la adición de componentes específicos y la traducción de los textos, por lo que también se le puede denominar regionalización"* (Diccionario Ilustrado Océano de la Lengua Española. Edición del Milenio).

Estos conceptos permiten tener un mejor entendimiento del problema a resolver, teniendo en cuenta que la localización y la internacionalización del software están estrechamente relacionados. Por ejemplo, la internacionalización permite usar distintos tipos de formatos de fecha, y la localización es seleccionar el formato adecuado para una región específica.

1.2 Proceso de internacionalización en los marcos de trabajo a nivel internacional

Numerosos son los marcos de trabajo que se utilizan en la construcción de aplicaciones web, que realizan el proceso de internacionalización adaptando el contenido a diferentes idiomas y regiones, por tal motivo se realizó una selección de los más usados, con el objetivo de analizar y describir los mecanismos de internacionalización que puedan ser utilizados en el desarrollo de la solución propuesta, seleccionando como referencia la encuesta realizada por Bruno Skvorc programador de Croacia con estudios de máster en Ciencias de la Computación, Inglés, Literatura y director de canal de PHP de SitePoint. Los resultados más detallados de la encuesta se pueden encontrar en el sitio web sitepoint (Skvorc, 2015). La misma generó la siguiente gráfica:

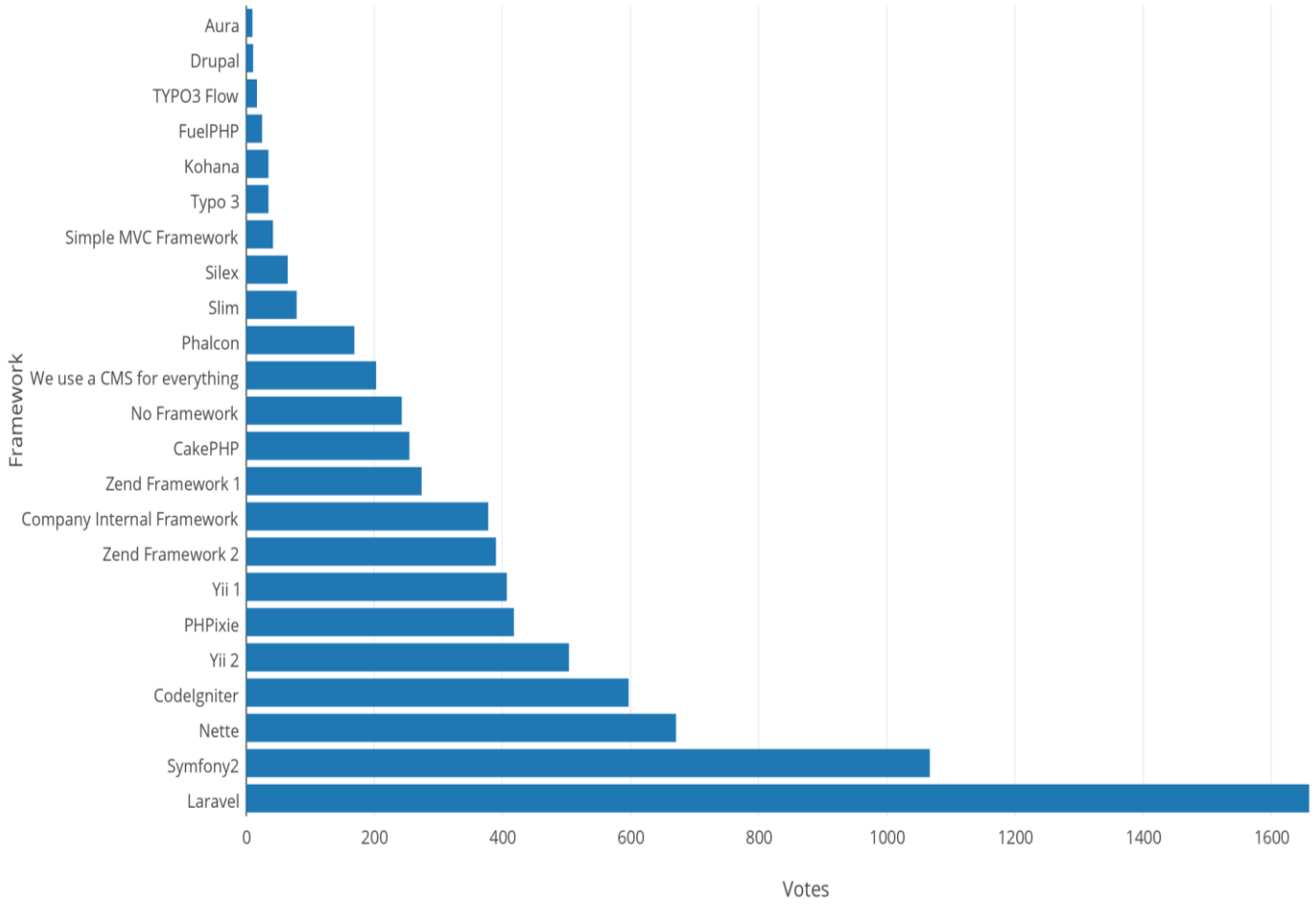


Figura 2. Gráfica de la encuesta realizada por Bruno Skvorc sobre los marcos de trabajo

A continuación se muestra el estudio de cómo las herramientas seleccionadas realizan el proceso de internacionalización.

Laravel

Laravel maneja una sintaxis expresiva, elegante, con el objetivo de eliminar la molestia del desarrollo web facilitando las tareas comunes, como la autenticación, enrutamiento, sesiones y caché. Proporciona, potentes herramientas accesibles necesarias para construir grandes aplicaciones robustas, con un contenedor de controles de inversión, sistema de migración expresiva, y el apoyo de las pruebas unitarias estrechamente integrada. Laravel se puede utilizar para aplicaciones de nivel empresarial enormes o simples usando la API JSON, lo que significa que es perfectamente adecuado para todos los tipos y tamaños de proyectos (Hostdime, 2014).

Internacionalización (I18N): Laravel ayuda a crear sistemas en diferentes idiomas de manera sencilla. Para poder lograr esto de la manera más escalable posible se deben tener todos los textos en archivos separados por cada idioma. Cada uno de estos archivos debe tener un arreglo con todos los textos en diferentes idiomas. Para utilizar los textos, Laravel tiene la función Lang::get(), la cual se llama donde se quiera imprimir alguno de los strings que se encuentran en los archivos de idiomas. El framework busca el texto automáticamente en el archivo del idioma que tenga establecido y si no lo consigue imprime el parámetro. Para cambiar el idioma se utiliza la función App::setLocale a la cual se le pasa como parámetro el idioma que se desee utilizar (Velásquez, 2014).

Symfony2

Diseñado con el objetivo de optimizar la creación de las aplicaciones web, con el uso de sus características. Posee una librería de clases que permiten reducir el tiempo de desarrollo.

Symfony está desarrollado en PHP5, se puede utilizar en plataformas *nix (Unix, Linux) y Windows. Requiere de una instalación, configuración y líneas de comando, incorpora el patrón MVC, soporta AJAX, plantillas y un gran número de bases de datos (Valdés, 2007).

Para traducir un mensaje, Symfony2 utiliza un proceso sencillo: (Eguiluz, 2014)

Symfony2 incluye tres opciones de configuración relacionadas con la internacionalización. La primera se define en el archivo *app/config/parameters.yml.* Esta opción es la más importante de todas, ya que su valor se utiliza en otras partes y opciones de configuración de la aplicación. Las otras dos opciones se configuran en el archivo *app/config/config.yml.* El tiempo que se tarda en comparar dos cadenas de texto depende de cuántas diferencias existan entre ellas.

CodeIgniter

CodeIgniter es un framework PHP de desarrollo de aplicaciones web. Su objetivo es permitir el desarrollo de proyectos mucho más rápido. Es de código abierto y utiliza la arquitectura MVC (Model-View-Controller). Su principal ventaja es ser sencillo y ligero todo empapado de Zend o Symfony con una curva de aprendizaje rápido (Technology, 2014).

Internacionalización (I18N): CodeIgniter Clases de idiomas ofrece funciones para recuperar archivos de idioma y líneas de texto para el propósito de la internacionalización. Los archivos de idioma se encuentran dentro del directorio de idioma. CodeIgniter busca automáticamente el directorio de idioma en dos lugares, primero dentro de subdirectorio de la aplicación "lenguas", y luego, si no se encuentra, dentro del subdirectorio de sistema de CodeIgniter. Cada idioma se almacena en su propio subdirectorio. Por

ejemplo, los archivos de idioma inglés se pueden almacenar en sistema / lenguaje / Inglés o application / lenguaje / Inglés (Helmut, 2013).

Yii -Yes It's

Yii es un framework de desarrollo de alto rendimiento, muy flexible y versátil, escrito en PHP5 para el desarrollo rápido de aplicaciones web. Yii es software libre liberado bajo una licencia BSD (Berkeley Software Distribution), y tiene la concepción de hacer las cosas de manera sencilla, elegante y rápidas, ayudando con esto a construir aplicaciones eficientes, que fácilmente pueden ser mantenidas y escaladas (leninmhs, 2013).

Internacionalización (I18N): Yii soporta traducción de mensajes, formatos de fecha y hora, formatos de números y la localización de interfaz de usuario, para un sistema que haga uso de varios idiomas (leninmhs, 2013).

Yii proporciona soporte para I18N en varios aspectos: (yiiframework, 2013)

- ✓ Proporciona los datos de localización para cada idioma posible y variante.
- ✓ Proporciona mensaje y el archivo de servicio de traducción.
- ✓ Proporciona fecha dependiente de la localización y el formato de tiempo.

La internacionalización de textos con Yii se puede hacer con diversas fuentes: (framework, 2013)

- ✓ Un array asociativo de claves/valores en un conjunto de directorios: esta es la opción predeterminada que utiliza la clase CPhpMessageSource.

Zend Framework

El marco de trabajo Zend contiene una impresionante matriz de funciones que pueden proveer desarrollo de aplicaciones a nivel corporativo, por lo que necesita un buen conocimiento de PHP (guiadehostingweb, 2012).

El Zend Frameworks es simple, no necesita instalación especial, requiere PHP5 e incorpora el patrón MVC. Utiliza el 100 % de código orientado a objetos y utiliza la mayor parte de las nuevas características de PHP 5.3, es decir, espacios de nombres, funciones y cierres.

La internacionalización en Zend Frameworks:(guiadehostingweb, 2012)

Zend Framework proporciona apoyo a I18n a través de una combinación de componentes, incluyendo Zend_Locale, Zend_Date, Zend_Measure, Zend_Translate, Zend_Currency y Zend_TimeSync.

En el análisis de las herramientas se realiza una comparación cualitativa teniendo en cuenta los siguientes criterios:

- ✓ Almacenamiento: define la estructura de almacenamiento utilizada para configurar el idioma.
- ✓ Directorio de acceso: especifica la localización del idioma en los marcos de trabajo analizados.
- ✓ Cambio de idioma: describe el mecanismo utilizado para realizar el cambio del idioma.

Tabla 1: Análisis de las herramientas existentes

Criterios	CodeIgniter	Yii -Yes It's	Symfony	Laravel	Zend Framework
Almacenamiento	Archivos de idioma	Conjunto de directorios	Archivos parameters.yml y *config.yml*	Archivos separados por cada idioma	Archivos .php
Directorio de acceso	application/language o en la carpeta global system/language	clase CPhpMessageSource	Variable especial llamada _locale	Función Lang::get()	application/confis/languages
Cambio de idioma	Mediante el archivo con las configuraciones del idioma especificado.	Mediante un array asociativo de claves/valores	Mediante catálogos, o archivos de texto en formato XLIFF, PHP o YAML	Mediante la función App::setLocale pasándole como parámetro el idioma	Componente Zend_Translate

1.3 Análisis de las herramientas existentes

En el análisis y descripción de los marcos de trabajo presentados se pudo fundamentar sobre el mecanismo utilizado para internacionalizar las aplicaciones que se construyen con los mismos. En el caso de **CodeIgniter** está provisto de una clase que provee las funcionalidades necesarias para que un sitio pueda soportar múltiples lenguajes, este crea los archivos de idioma del core del marco de trabajo que se almacenan en el directorio system/lenguaje, por ejemplo, en el caso del idioma inglés quedaría

system/lenguaje/english, en su ejecución CodeIgniter busca en el directorio application/language, si el directorio o lenguaje especificado no existe entonces buscará en la carpeta global system/language, carga el archivo con las configuraciones del idioma especificado, ejemplo $autoload[‘language’] = array(‘formulario_lang’), formulario_lang’ es el archivo de extensión .php que contiene todas las etiquetas con sus respectivos títulos, ejemplo $lang[form_idioma_esp]=”Español”. El análisis de esta herramienta sirvió para identificar la forma de ejecución del idioma, aunque lo hacen a través de una clase .php, el mecanismo es similar al que provee el marco de trabajo Sauxe, lo que este último, lo realiza por medio de un fichero .json directamente del javascript, no del servidor como lo crea CodeIgniter.

El análisis de la herramienta **Yii -Yes It’s** permitió brindar elementos claves que se pueden ejecutar en la construcción de la solución propuesta, como un array asociativo de claves/valores en un conjunto de directorios, en este caso **Yii -Yes It’s** a partir de palabras claves permiten traducir para un idioma especificado, ejemplo array (‘message’ => ‘mensaje’). En la solución propuesta se pueden guardar las direcciones donde se encuentran los archivos de los idiomas y a través de ellas se pueden mostrar y ejecutar estos archivos.

El análisis de las herramientas **Symfony2 y Laravel** sirvieron para identificar la forma de ejecución del idioma en los mismos, en este caso Symfony2 lo realiza a través del fichero *app/config/config.yml* se pueden configurar las rutas de acceso para cada uno de los idiomas especificados. Laravel por su parte utiliza un conjunto de archivos y cada uno de estos archivos debe tener un arreglo con todos los textos en diferentes idiomas. Para cambiar el idioma se utiliza la función App:setLocale a la cual se le pasa como parámetro el idioma que se desee utilizar.

Zend Framework es uno de los marcos de trabajo más importantes de los estudiados porque precisamente el mecanismo de configuración del idioma es el que se utiliza en el marco de trabajo Sauxe, consecuente a que Sauxe está construido sobre esta aplicación. En este caso se registran todos los idiomas en la bases de datos, con sus respectivos identificadores y la localización donde se encuentra la información a mostrar en un idioma especificado, se crea una función que permite conocer el id del idioma que trae por defecto el formulario y transforma el formulario estableciendo el nuevo idioma.

Por último, haciendo un análisis más concreto se pudo apreciar que los seis marcos de trabajo analizados realizan el proceso de internacionalización de diferentes formas. Pero, a pesar de no cumplir con todos los requisitos que se desea integrar a Sauxe, estas herramientas brindan ideas que pueden ser empleadas en el desarrollo de la nueva solución.

1.4 Metodología de desarrollo

Metodología a utilizar

En la Universidad de las Ciencias Informáticas (UCI) se utiliza para el desarrollo de sus productos de software la Metodología de desarrollo para la Actividad productiva de la UCI, basada en una variación de la metodología ¨Proceso Unificado Ágil¨(AUP por sus siglas en inglés) en unión con el modelo CMMI-DEV v 1.3.

La nueva metodología cuenta con tres fases de desarrollo entre las que se encuentran: Inicio, Ejecución y Cierre, para lograr el objetivo de la presente investigación se utilizan determinados artefactos de las disciplinas ingenieriles de la fase Ejecución. A continuación se muestran los productos de trabajo desarrollados por cada una de estas disciplinas.

Disciplinas Variación AUP-UCI

1. **Modelado de negocio:** se realiza el modelo del dominio representado a través del Modelo conceptual.
2. **Requisitos:** en esta disciplina se identifican los requisitos funcionales y no funcionales. Se generan como productos de trabajo las Historias de usuario por cada uno de los requisitos definidos, la Evaluación de requisitos y la Matriz de trazabilidad de requisitos.
3. **Análisis y diseño:** en esta disciplina se modela el sistema a través del Diagrama de clases.
4. **Implementación:** en la implementación, a partir de los resultados del Análisis y Diseño se construye el componente Internacionalización.
5. **Pruebas interna:** en esta disciplina se le realizan pruebas al software. En el caso de la solución planteada solo se generarán Diseños de casos de prueba y un Manual de usuario.
6. **Pruebas de aceptación:** se realizan por parte de los profesionales del departamento, con el fin de identificar no conformidades en el sistema y en la documentación.

1.5 Herramientas y tecnologías para el desarrollo

Una correcta selección de las herramientas y tecnologías a utilizar permite en gran medida el desarrollo exitoso de los sistemas informáticos, las definidas para desarrollar el componente Internacionalización son las siguientes:

1.5.1 Herramientas CASE

Visual Paradigm 8.0

Visual Paradigm es una herramienta CASE (Computer Aided Software Engineering o Ingeniería de Software Asistida por Computadora) profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Esta herramienta ayuda a construir aplicaciones con calidad y con un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y documentación. La herramienta CASE también proporciona abundantes tutoriales, demostraciones interactivas y proyectos en UML (Unified Modeling Language o Lenguaje Unificado de Modelado). Para los ingenieros y arquitectos de software es excelente ya que permite centralizar y planificar las ideas, además de mejorar la productividad en el desarrollo y mantenimiento del software, aumentar la calidad del mismo, reducir el tiempo de desarrollo y mantenimiento, mejorar la planificación de un proyecto y realizar una gestión global en todas las fases de desarrollo de software con una misma herramienta (K. Prada Nicot, 2009).

1.5.2 Herramientas para el desarrollo colaborativo

Subversion 1.6.17

Subversion es un sistema centralizado de control de versiones de código abierto. Se caracteriza por su fiabilidad como un refugio seguro para los datos valiosos, la simplicidad de su modelo, uso y capacidad para apoyar las necesidades de una amplia variedad de usuarios y proyectos, desde proyectos particulares hasta operaciones empresariales a gran escala (Apache, 2014).

1.5.3 Herramientas para el desarrollo

PostgreSQL 9.0

Es un gestor de base de datos relacional y libre. Tiene soporte total para transacciones, disparadores, vistas, procedimientos almacenados y almacenamiento de grandes objetos. Se destaca en ejecutar consultas complejas, consultas sobre vistas, subconsultas y JOINSv de gran tamaño. Permite la definición de tipos de datos personalizados e incluye un modelo de seguridad completo. Como toda herramienta de software libre PostgreSQL tiene, entre otras ventajas, las de contar con una gran comunidad de desarrollo en Internet, su código fuente está disponible sin costo alguno y algo muy importante es que dicha herramienta es multiplataforma. Además de sus ofertas de soporte, cuenta con una importante comunidad de desarrollo de los que se pueden obtener beneficios (PostgreSQL, 2008).

NetBeans 8.0

IDE NetBeans es un entorno de desarrollo integrado de código abierto para desarrolladores de software. Cuenta con todas las herramientas necesarias para crear aplicaciones profesionales de escritorio, empresariales, web y aplicaciones móviles con la plataforma Java, así como con C / C++, PHP, JavaScript y Groovy. NetBeans recibe el soporte integrado para lenguajes dinámicos, todo en una herramienta de gran alcance. El editor de PHP de NetBeans ofrece plantillas de código y generación (getters y setters), la refactorización, información sobre herramientas de parámetros, consejos y soluciones rápidas, y la finalización de código inteligente (Oracle, 2012).

Apache 2.2

El servidor web Apache 2.2 está desarrollado bajo el criterio Open Source. Funciona en múltiples sistemas operativos, lo que lo hace prácticamente universal. Es un servidor altamente configurable de diseño modular. Entre sus principales características se encuentran: (TheApache, 2013)

- ✓ Multiplataforma.
- ✓ Presenta una alta configuración en la creación y gestión de logs. Apache permite la creación de ficheros de log a medida del administrador, de este modo se puede tener un mayor control sobre lo que sucede en el servidor.
- ✓ Se desarrolla de forma abierta.

PgAdmin III

Es una herramienta de código abierto para la administración de bases de datos PostgreSQL y derivados. Incluye una interfaz administrativa gráfica y una herramienta de consulta SQL. Es multiplataforma, libre y se puede adquirir el código fuente desde la web oficial. Permite desde ejecución de consultas SQL simples, hasta la elaboración de bases de datos complejas, con el apoyo de las últimas características de PostgreSQL. No requiere ningún controlador adicional para comunicarse con el servidor de base de datos. La aplicación se encuentra desarrollada por una comunidad de especialistas en base de datos de todo el mundo y se encuentra disponible en más de 30 idiomas (PgAdmin 2011).

1.5.4 Librerías y marcos de trabajo

Sauxe 2.3

Sauxe es un marco de trabajo que contiene un conjunto de componentes reutilizables que provee la estructura genérica para el desarrollo de aplicaciones web de gestión, logrando una mayor estandarización, flexibilidad, integración y agilidad en el proceso de desarrollo, siguiendo el paradigma de independencia tecnológica por el cual apuesta el país.

El marco de trabajo Sauxe da solución a un sin número de escenarios o aspectos arquitectónicos como: gestión y configuración dinámica de caché, integración de componentes de forma distribuida o no distribuida, acceso a bases de datos a través de una capa de abstracción, gestión de concurrencia de recursos, administración centralizada de transacciones, gestión dinámica de las trazas generadas por los sistemas, implementación de mecanismos de autenticación y autorización, implementación de mecanismos de mensajería y control de excepciones, gestión y configuración dinámica de precondiciones, postcondiciones y validaciones de variables, gestión y configuración de flujos de trabajo, visualización de las funcionalidades de un sistema, entre otros escenarios de alta complejidad, garantizando los atributos de calidad de los sistemas que se desarrollen con el mismo. Cuenta con una arquitectura en capas que a su vez presenta en su capa superior un MVC (Modelo-Vista-Controlador) (Baryolo, 2010).

ExtJS 4.2.1

Proporciona una nueva arquitectura de aplicación que no sólo organiza su código, sino que también reduce la cantidad de código fuente que se debe de escribir. La nueva arquitectura sigue un patrón M-V-C (componentes, 2014).

Zend Framework 1.11

Se trata de un framework para desarrollo de aplicaciones y servicios web con PHP que brinda soluciones para construir sitios web modernos, robustos y seguros. Además es de código abierto y trabaja con PHP5. A diferencia de CakePHP que trabaja con PHP4 y PHP5, este framework está formado por una serie de métodos estáticos y componentes que usarán estos métodos. Los componentes son varios y variados y aunque alguno es posible que no se use nunca, hay otros que pueden ser usados hasta la saciedad, por ejemplo el componente para la BD (Esser, 2009).

Doctrine 1.2.2

Es un mapeador de objetos-relacional (ORM) escrito en PHP, es una capa de abstracción que se sitúa justo encima de un Sistema Gestor de Base Datos. Está completamente diseñado utilizando la técnica orientada a objetos, ofreciendo una capa de persistencia transparente a los objetos de PHP. Uno de sus puntos fuertes es que cuenta con su propio lenguaje de consultas a bases de datos llamado DQL (Doctrine 2014).

1.5.6 Lenguaje de Modelado

UML

El Lenguaje Unificado de Modelado (UML) es un leguaje de modelado estandarizado de propósito general en el campo de la ingeniería de software orientada a objetos. UML incluye un conjunto de técnicas de notación gráfica para crear modelos visuales de programación orientada a objetos (Pressman, 2005).

UML representa para los desarrolladores de aplicaciones y sistemas una serie de ventajas, al igual que para las organizaciones, entre estos beneficios destacan: (ENTEL, 2013)

- ✓ Produce un aumento en la calidad del desarrollo.
- ✓ Mejora en un 50% o más los tiempos totales de desarrollo.
- ✓ Brinda la posibilidad de obtener un "plano" del sistema.
- ✓ Ofrece un mejor soporte a la planificación y control del proyecto.
- ✓ Constituye un lenguaje de modelado entendido tanto por humanos como por máquinas.
- ✓ Permite realizar una verificación y validación del modelo realizado.
- ✓ Se pueden automatizar determinados procesos y permite generar código a partir de los modelos y viceversa.

1.5.7 Lenguajes de programación

Programación del lado del servidor

PHP5.3

PHP es un lenguaje interpretado de alto nivel, especialmente pensado para desarrollos web y el cual puede ser incrustado en páginas HTML. La mayoría de su sintaxis es similar a los lenguajes C, Java y Perl y es fácil de aprender. La meta de este lenguaje es permitir escribir a los programadores páginas web más dinámicas de una manera rápida y fácil (Gutmans, y otros, 2004). Lo mejor de usar PHP es que es extremadamente simple para el principiante, pero a su vez, ofrece muchas características avanzadas para los programadores profesionales (M. A. Friedhelm Betz, 2012).

Programación del lado del cliente

JavaScript 1.3

JavaScript es un lenguaje de programación interpretado, utilizado principalmente en su forma del lado del cliente, permitiendo mejoras en la interfaz de usuario sobre todo en páginas web dinámicas. Es un lenguaje interpretado que la mayoría de los navegadores como Internet Explorer, Netscape, Opera, Mozilla Firefox, entre otros, lo soportan (ECMA, 2010)**.**

1.6 Arquitectura de software

Según el estándar IEEE (Institute of Electrical and Electronics Engineers): “la Arquitectura de Software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución”.

La arquitectura del software de un programa o sistema de cómputo es la estructura o las estructuras del sistema, que incluyen los componentes del software, las propiedades visibles externamente de esos componentes y las relaciones entre ellos. Es una representación que permite analizar la efectividad del diseño para cumplir con los requisitos establecidos, considerar opciones arquitectónicas en una etapa en que aún resulta relativamente fácil hacer cambios al diseño y reducir los riesgos asociados a la construcción del software.

En la disciplina arquitectura de software se determina el estilo arquitectónico y el patrón arquitectónico, cuyas definiciones se muestran a continuación:

Estilo arquitectónico: expresa la arquitectura en el sentido más formal y teórico, describiendo una clase de arquitectura, o piezas identificables de las arquitecturas empíricamente dadas. Esas piezas se encuentran repetidamente en la práctica, trasuntando la existencia de decisiones estructurales coherentes. Una vez que se han identificado los estilos, es lógico y natural pensar en reutilizarlos en situaciones semejantes que se presenten en el futuro (Reynoso, 2004).

Patrón arquitectónico: expresa el esquema de organización estructural fundamental para sistemas de software. Este esquema provee un conjunto de subsistemas predefinidos, especifica sus responsabilidades e incluye reglas y pautas para la organización de las relaciones entre ellos. Pudiera representarse como una plantilla para arquitecturas de software concretas, que especifica las propiedades estructurales de una aplicación (Buschmann, 2007).

Conclusiones del capítulo

En el capítulo concluido se realizó un estudio de los conceptos fundamentales relacionados con la situación problemática, permitiendo tener un mejor dominio del problema en cuestión.

Además se evidencia en el análisis y descripción de los marcos de trabajo que realizan el proceso de internacionalización, características comunes en la realización de la configuración del idioma, brindando elementos que pueden ser empleados en el desarrollo de la nueva solución, como la generación de las rutas de los ficheros del idioma, la implementación y ejecución de los cambios del idioma por defecto al nuevo idioma. Se hace alusión al marco de trabajo **Zend Framework** que es el que permite realizar las configuraciones del idioma en Sauxe, a través de un mecanismo del lado del servidor.

Se identifican las disciplinas de la Metodología de desarrollo para la Actividad productiva de la UCI que van a guiar el proceso de desarrollo de software en su variación propuesta en la universidad, quedando definido solamente realizar las disciplinas de Requisitos, Análisis y Diseño, Implementación, Pruebas internas y Pruebas de aceptación.

CAPÍTULO II: PROPUESTA DE SOLUCIÓN

Introducción

En este capítulo se realiza una propuesta de solución que describe el componente Internacionalización para los sistemas que se construyen sobre el marco de trabajo Sauxe .Se generan los artefactos por cada una de las disciplinas del proceso de desarrollo de software, propuestos en la Metodología de desarrollo para la Actividad productiva en la UCI. Se realiza la descripción del Modelo conceptual definido para la solución. En la disciplina de Requisitos se describen las técnicas de captura de requisitos establecidas y aplicadas, se realiza la administración y validación de los requisitos y se aplican técnicas de agrupación de requisitos. En la disciplina de Análisis y Diseño se define la arquitectura donde se identifican los patrones y estilos arquitectónicos utilizados, se presentan el modelo conceptual de la aplicación, el modelo de diseño y el modelo de datos. Se muestran los prototipos de interfaz de usuario asociados a cada uno de los requisitos funcionales.

2.1 Modelado de negocio

Para el sistema que se desea desarrollar no existen procesos de negocio definidos, por lo que no se realiza una descripción del proceso de negocio. Para el modelo de dominio se genera el modelo conceptual el cual es la representación visual de los conceptos u objetos del mundo real que son significativos para el problema, así como la relación existente entre ellos (Sommerville, 2006).

2.1.1 Descripción del Modelo conceptual

El Modelo conceptual que se propone en la figura 3, muestra los conceptos asociados a la investigación, representado por el concepto de Sauxe como marco de trabajo, que tiene asociado a Acaxia como Sistema de Gestión Integral, brindando la seguridad del mismo. Este último, incluye el componente Internacionalización, que contiene los conceptos de calendario, hora, zona horaria e idioma con sus respectivos atributos.

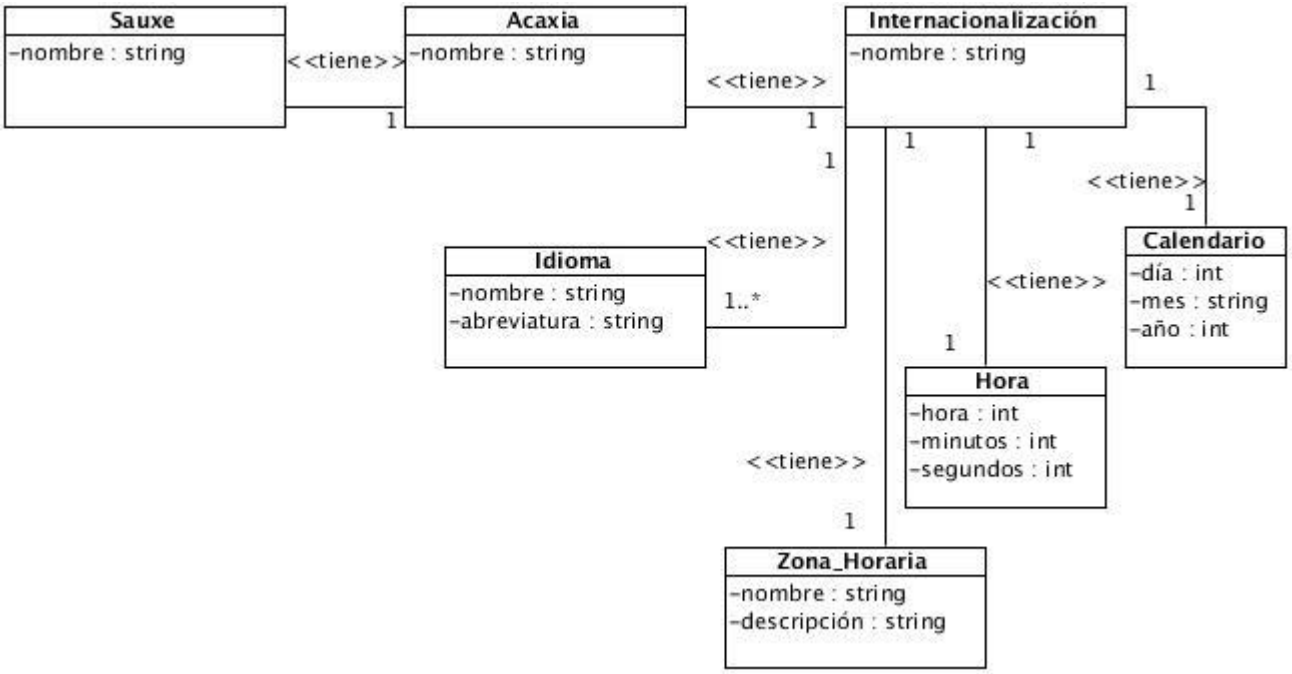


Figura 3.Modelo conceptual

2.2 Requisitos de software

Para el proceso de captura de los requisitos del componente se utilizaron las técnicas siguientes:

- ✓ **Entrevistas:** el método se aplicó a través de entrevistas abiertas a especialistas del proyecto que realizan las construcciones de interfaces y al jefe de proyecto que proporcionó la información de las necesidades del marco de trabajo Sauxe en cuestiones de internacionalización.
- ✓ **Prototipos:** el método se aplicó en la construcción de los prototipos de interfaces de usuarios, con la librería de ExtJS se personalizaron las interfaces.
- ✓ **Tormenta de ideas:** el método se aplicó en los talleres con los tutores de la tesis, donde se analizaron los elementos asociados a la internacionalización.
- ✓ **Observación:** el método se aplicó en el análisis de las herramientas existentes, se realizó una observación de cómo diferentes marcos de trabajo realizaban la configuración de los idiomas y cómo se podrían ejecutar en Sauxe.

2.3 Requisitos funcionales

Los requisitos funcionales son las capacidades o condiciones que el sistema debe cumplir, para que el usuario solucione un problema o cumpla sus objetivos. Deben estar claros y libres de ambigüedades, asegurando que los involucrados comprendan claramente el significado de cada uno. Teniendo en cuenta

el escenario cuatro de la Metodología de desarrollo para la Actividad productiva de la UCI, en la solución se identifican requisitos pero se describen historias de usuario. A continuación se muestran los requisitos funcionales para el componente Internacionalización a desarrollar.

Agrupación de requisitos

Luego de aplicado la técnica de captura de requisitos fueron identificados 18 requisitos funcionales, agrupados en 5 grupos, aplicando el criterio de agrupación funcional que permite agrupar requisitos que tienen una relación funcional directa, según el tipo de datos que van a manejar o según los conceptos que van a cubrir, quedando agrupados de la siguiente forma:

Tabla 2. Listado de requisitos funcionales

Agrupación de requisitos	Nombre del requisito
Gestionar idioma	RF 1.1 Adicionar idioma
	RF 1.2 Modificar idioma
	RF 1.3 Eliminar idioma
	RF 1.4 Buscar idioma
Configurar fichero de idioma del componente	RF 2.1 Cargar fichero de idioma del componente
	RF 2.2 Modificar fichero de idioma del componente
	RF 2.3 Restablecer fichero de idioma del componente
Configurar zona horaria	RF 3.1 Mostrar zona horaria
	RF 3.2 Modificar zona horaria
Calendario	RF 4.1 Mostrar formato de la fecha
	RF 4.2 Mostrar formato de la hora
	RF 4.2 Configurar formato de la hora
	RF 4.3 Configurar formato de la fecha
	RF 4.4 Mostrar calendario
	RF 5.1 Listar funcionalidades
	RF 6.1 Listar subsistemas
Configurar fichero idioma de la librería	RF 7.1 Cargar fichero de idioma de la librería
	RF 7.2 Configurar fichero de idioma de la librería

A continuación se muestra una propuesta de una historia de usuario, en este caso del requisito: Adicionar idioma, las restantes se encuentran en los productos de trabajo.

Tabla 3 .Historia de usuario del requisito Adicionar idioma

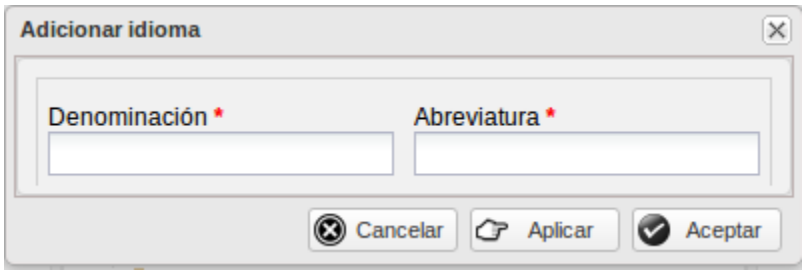
Número: RF1.1	**Nombre del requisito:** Adicionar idioma
Programador: Yunior Ismael Charro Pérez	**Iteración asignada:** 1
Prioridad: Media	**Tiempo estimado:** 3 días
Riesgo en desarrollo: Desconocimiento de los términos del idioma.	**Tiempo real:** 1 semana
Descripción: Permite insertar un nuevo idioma al marco de trabajo Sauxe, además de que se genera un fichero de idioma en todos los componentes del marco de trabajo.	
Observaciones: N/A	
Prototipo de interfaz: 	

Figura 4.Prototipo de interfaz de usuario Adicionar idioma

2.3.1 Administración de requisitos

Según Sommervelle I.; 2005. “Ingeniería de Software” plantea que: La administración de requisitos es el proceso de comprender y controlar los cambios en los requisitos del sistema.

Los requisitos a desarrollar y los cambios introducidos durante el ciclo de vida deben ser entendidos, identificados, controlados y traceados para contribuir al correcto desarrollo del producto final. Sus objetivos son:

- ✓ Controlar los cambios en la línea base de los requisitos.
- ✓ Mantener la inconsistencia entre los planes del proyecto y los requisitos.
- ✓ Controlar las versiones de los requisitos individuales y de las especificaciones de requisitos de software.

- ✓ Mantener el estado de los requisitos en la línea base.
- ✓ Mantener la traza entre los requisitos.

Actividades desarrolladas en la Administración de Requisitos

El componente Internacionalización se desarrolló en el departamento de Desarrollo de Componentes, y teniendo en cuenta que el resultado de esta investigación es parte de un proyecto aún no oficializado, no se han realizado todas las actividades definidas en el proceso de administración de requisitos. A continuación se describen las principales evidencias que se necesitan para realizar la administración de los requisitos:

De los artefactos propuestos, por su utilidad e importancia en el marco del trabajo se mantuvo la elaboración de la matriz de trazabilidad, la cual fue realizada con la herramienta Visual Paradigm y se muestra a continuación. En la Tabla 5 se muestra la cantidad de no conformidades detectadas en la aplicación y la documentación por cada iteración, teniendo un total de 10 no conformidades significativas y cinco no significativas.

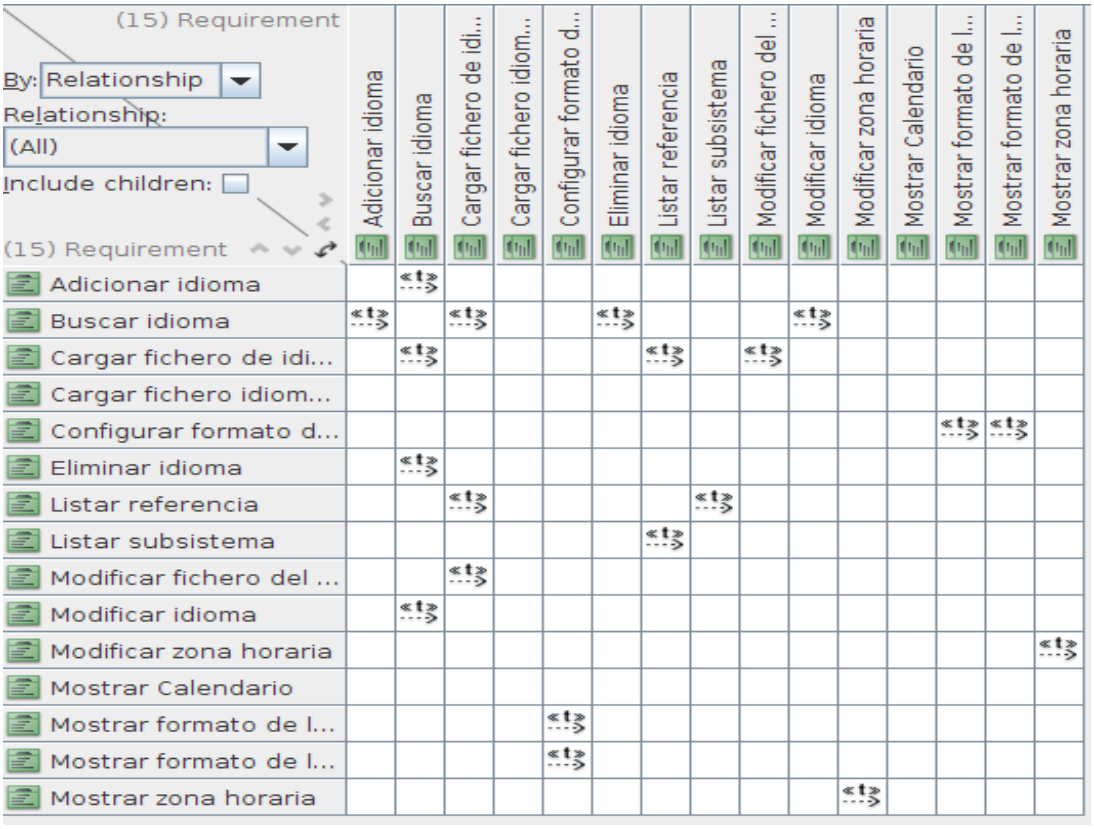


Figura 5. Matriz de trazabilidad de requisitos

2.3.2 Validación de los requisitos

Este proceso tiene por finalidad comprobar que los requisitos del software poseen todos los atributos de calidad los cuales pueden ser consistentes, completos, precisos, realistas, verificables y definen lo que el usuario desea del producto final. La realización de estas actividades en este momento pretende evitar los altos costos que significaría el tener que corregir una vez avanzado el desarrollo (Software, 2014).

La validación de los requisitos se realizará a través de diversos métodos en los cuales se encuentran:

Revisión de requisitos: las revisiones de requisitos se realizaron por parte de los especialistas del departamento.

Prototipado: en este método se realizaron prototipos funcionales, en los cuales se implementaron algunas funciones, que a medida que se construían, se comprobaban que eran las apropiadas, se corregían, se refinaban, y se añadían otras.

Generación de casos de prueba (test de requisitos): en la solución se realizaron 13 diseños de casos de prueba que permitieron verificar el cumplimiento de los requisitos funcionales.

Creación de manuales de usuario: en este método se realizó un manual de usuario con el objetivo de brindar las instrucciones necesarias para que un usuario pueda utilizar sin problemas el componente Internacionalización.

2.4 Requisitos no funcionales

Los requisitos no funcionales son las propiedades o cualidades que el producto debe tener, especificando criterios que pueden usarse para juzgar la operación de un sistema (Sommerville, 2006). El desarrollo del componente para la internacionalización forma parte del marco de trabajo Sauxe, por lo que los requisitos no funcionales a los que se debe acoger la aplicación a desarrollar fueron los establecidos por el centro CEIGE, los mismos se mencionan a continuación:

Software

Se requiere para las PC clientes:

- ✓ Navegador Mozilla Firefox 29.0 o superior.
- ✓ Sistemas operativos GNU/Linux, Windows XP o superior.

Se requiere para las PC servidoras:

- ✓ Sistema operativo Linux en cualquiera de sus distribuciones.
- ✓ Servidor web Apache en su versión 2.0 o superior, con los módulos de PHP 5.0 disponibles.
- ✓ PostgreSQL en su versión 8.3 o superior, como Sistema Gestor de Base de Datos.

Hardware

En el caso de las PC clientes donde se ejecutará la aplicación, se requiere de las siguientes condiciones: Procesador Pentium IV 2x2 caché, con velocidad de procesamiento a 1 GHz como mínimo y 512Mb de RAM como mínimo.

En dependencia de la funcionalidad concebida para cada uno de los servidores, son las condiciones que se requiere que deban cumplir:

Servidor web: requiere un procesador Pentium IV 2X2 caché, velocidad del procesador de 1GHz como mínimo y memoria de 1GB de RAM.

Servidor de datos: requiere un procesador Pentium IV 2x2 caché, velocidad del procesador de 1 GHz como mínimo, memoria de 1 GB de RAM, almacenamiento en disco con una capacidad igual o superior a los 10 GB.

Rendimiento

El rendimiento del software en tiempo de ejecución para los escenarios Cargar idioma, Adicionar idioma, Modificar idioma y Eliminar idioma fue de 6.6 segundos, ver en el Anexo 8.

2.5 Modelo de diseño

El modelo de diseño es "una abstracción del modelo de implementación y su código fuente, el cual fundamentalmente se emplea para representar y documentar su diseño" (Baryolo, 2010). El mismo puede contener: diagramas, clases, paquetes, interfaces, entre otros, que son utilizados como entrada esencial en las actividades relacionadas a la implementación. En el presente epígrafe se muestran los elementos que se tuvieron en cuenta para el diseño de la solución, así como los resultados y artefactos generados durante esta parte del proceso de desarrollo.

2.5.1 Arquitectura de Software

Estilo Arquitectónico

Para el desarrollo del marco de trabajo Sauxe, en el departamento de Desarrollo de Componentes se definió el **Estilo N capas**, con cuatro capas fundamentales que se describen a continuación: (Baryolo, 2010)

- ✓ **Capa de presentación:** esta es la capa encargada de elaborar y mostrar las interfaces de los usuarios. La capa de presentación está compuesta principalmente por el marco de trabajo ExtJS y centra su desarrollo en tres componentes fundamentales archivos JS, archivos CSS, páginas clientes y páginas servidoras.

- ✓ **Capa de control o negocio:** en esta capa se encuentra ubicado el marco de trabajo Zend, el cual implementa el patrón Modelo-Vista-Controlador.
- ✓ **Capa de acceso a datos:** está ubicado el marco de trabajo Doctrine, como ORM para la comunicación con el servidor de datos mediante el protocolo de conexión a base de datos basado en objetos (PDO).
- ✓ **Capa datos:** en esta capa se encuentran ubicado los servidores de base de datos y los archivos XML donde se almacena la información de la aplicación.

Patrón arquitectónico

En la capa de control o negocio de la arquitectura descrita anteriormente para Sauxe, se implementa el patrón arquitectónico Modelo-Vista-Controlador (MVC), el cual se encarga de separar los datos de la aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos: (Baryolo, 2010)

- ✓ **Modelo:** está compuesto por datos, reglas de negocio y las funcionalidades correspondientes para la comunicación con el encargado de persistir los datos Doctrine.
- ✓ **Vista:** muestra la información del modelo al usuario.
- ✓ **Controlador:** gestiona las entradas y las respuestas del sistema al usuario.

2.6 Patrones de diseño

GRASP

En el diseño de la propuesta de solución se aplican los patrones GRASP, los cuales son un acrónimo que significa General Responsibility Asignment Software Patterns (patrones generales de software para asignar responsabilidades). A continuación se mencionan los patrones GRASP utilizados, que permite describir los principios fundamentales de diseño de objetos para la asignación de responsabilidades.

Experto: se aplica para asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad. En este caso, se refleja en la clase NomIdiomaModel; la cual contiene la información que forma parte de los valores de entrada de la solicitud que realiza la clase gestnomidioma a la GestnomidiomaController y viceversa. El uso de este patrón permite que se conserve el encapsulamiento, ya que los objetos se valen de su propia información para hacer lo que se le pide. Esto soporta un bajo acoplamiento, lo que favorece al hecho de tener sistemas más robustos y de fácil mantenimiento.

Creador: se aplica para la asignación de responsabilidades a las clases relacionadas con la creación de objetos, de forma tal que una instancia de un objeto sólo pueda ser creada por el objeto que contiene la información necesaria para ello. En este caso, el patrón se refleja en la clase GestnomidiomaController,

encargado de crear una instancia de la clase NomIdiomaModel y de gestionar el idioma en el componente Internacionalización.

Bajo Acoplamiento: el acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y con que recurre a ellas. En este caso, se refleja el bajo acoplamiento, en la clase GestnomidiomaController con el objetivo de que una clase no dependa de muchas clases, de esta forma, no se afectan las clases por cambios de otros componentes, son fáciles de entender por separado y fáciles de reutilizar.

Alta Cohesión: la cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. En este caso, se garantiza que la clase GestnomidiomaController, posea alta cohesión, de manera que las clases contengan las responsabilidades estrechamente relacionadas y que no realicen un trabajo enorme. El uso de este patrón permite que se pueda mejorar la claridad y facilidad con que se entiende el diseño, se simplifique el mantenimiento y las mejoras de funcionalidad.

Controlador: un controlador es un objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema. Define además el método de su operación. En este caso, se encuentra reflejado en la clase GestnomidiomaController, encargada del comportamiento del idioma y gestiona todos los eventos que ocurren en el mismo.

GoF

Los patrones de diseño Gang of Four (GoF) o grupo de los cuatro, se clasifican en tres grandes categorías: creacionales, estructurales y de comportamiento, para un total de 23 patrones. Para el desarrollo del componente Internacionalización se decidió utilizar un patrón perteneciente al grupo de los creacionales, el mismo se muestra a continuación.

Singleton: garantiza que una clase sólo tiene una única instancia, proporcionando un punto de acceso global a la misma. Se usa cuando debe haber únicamente una instancia de una clase y debe ser claro su acceso para los usuarios. Los mismos deben poder usar la instancia extendida sin modificar su código (Pérez Mariñán, 2007). El uso de este patrón se evidencia en las variables globales definidas en el marco de trabajo Sauxe y que son utilizadas por el componente Internacionalización.

2.7 Diagrama de clases del diseño con estereotipos web

Los diagramas de clases especifican las diferentes clases que serán utilizadas en el sistema, así como las relaciones que existen entre ellas.

El diagrama de clases muestra un conjunto de clases, interfaces y colaboraciones y las relaciones entre éstos. Los diagramas de clases muestran el diseño de un sistema desde un punto de vista estático; un

diagrama estático es el que muestra una colección de elementos (estáticos) declarativos (Grady Booch). En la figura 6 se muestra el diagrama de clases del diseño de la interfaz “Gestionar idiomas”, representado por la clase servidora **GestnomidiomaController,** que se encarga de construir o generar el resultado HTML[1] en gestnomidioma.phtml que conforma el código cliente **<<build>>**, el formulario componente envía los datos al código servidor para ser procesados los pedidos **<<submit>>**, y además forman parte del código cliente o resultado HTML, el código servidor solicita y envía la información a través de la clase NomIdiomaModel que es la encargada de almacenar la información del componente idioma. En la construcción de los formularios se utiliza la librería ExtJs en su versión 4.2.1, a través de la clase gestnomidioma.js se construyen las interfaces del componente Internacionalización y Gestjson.js la interfaz de los ficheros de idioma. Los restantes diagramas de clases del diseño se encuentran en el Anexo 3 de la versión digital del documento.

[1] **HTML:** por sus siglas en inglés **H**yper **T**ext **M**arkup **L**anguage

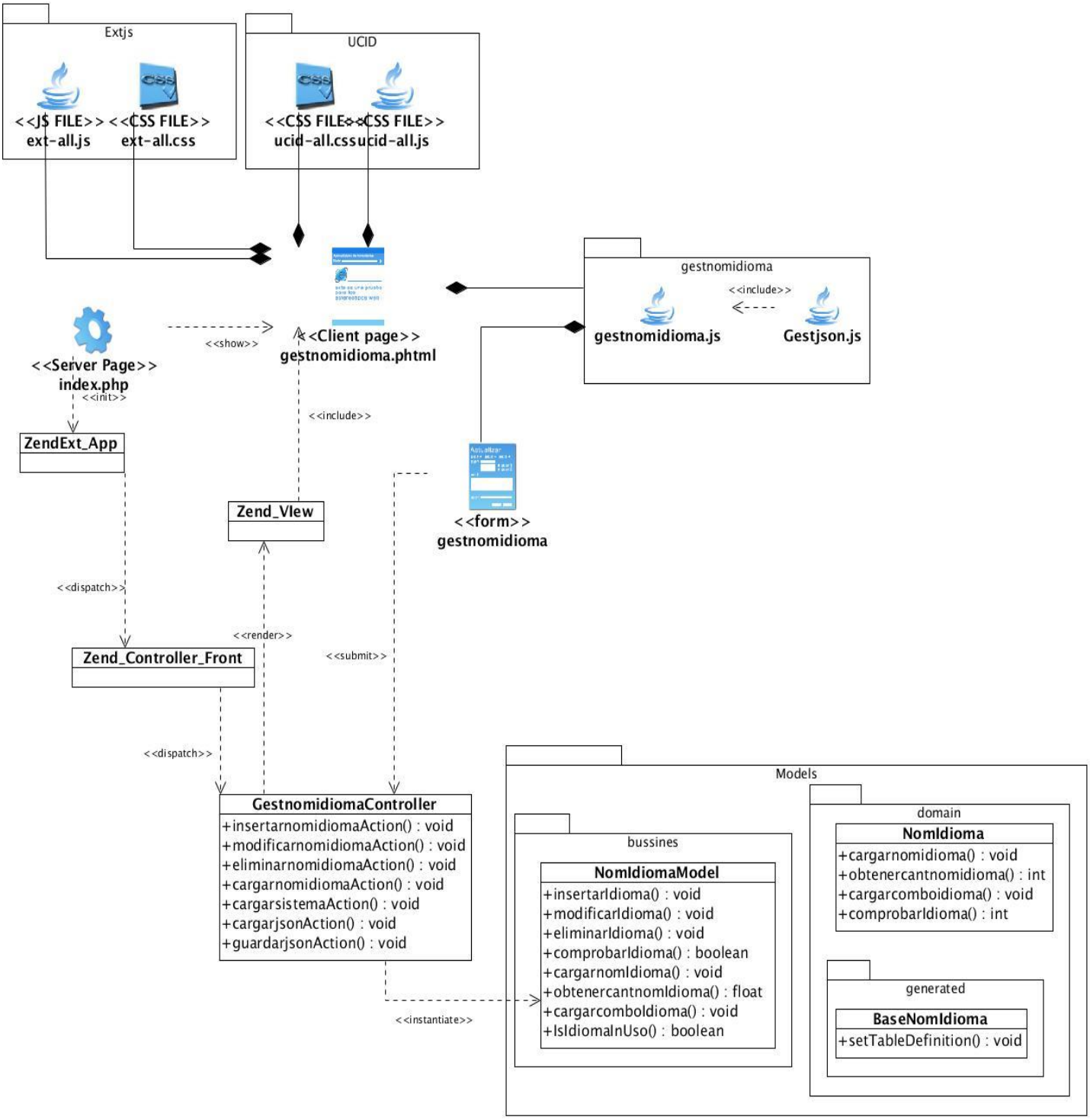


Figura 6. Diagrama de Clases del Diseño del requisito Gestionar Idioma

2.8 Modelo de datos

El Modelo de datos es una definición lógica, independiente y abstracta de los objetos, operadores y demás que en conjunto constituyen la máquina abstracta con la que interactúan los usuarios. Los objetos permiten modelar la estructura de los datos y los operadores permiten modelar su comportamiento.

El Modelo de datos definido en la solución propuesta es orientado a objetos y contiene 2 entidades persistentes. A continuación se muestra el diagrama que lo describe:

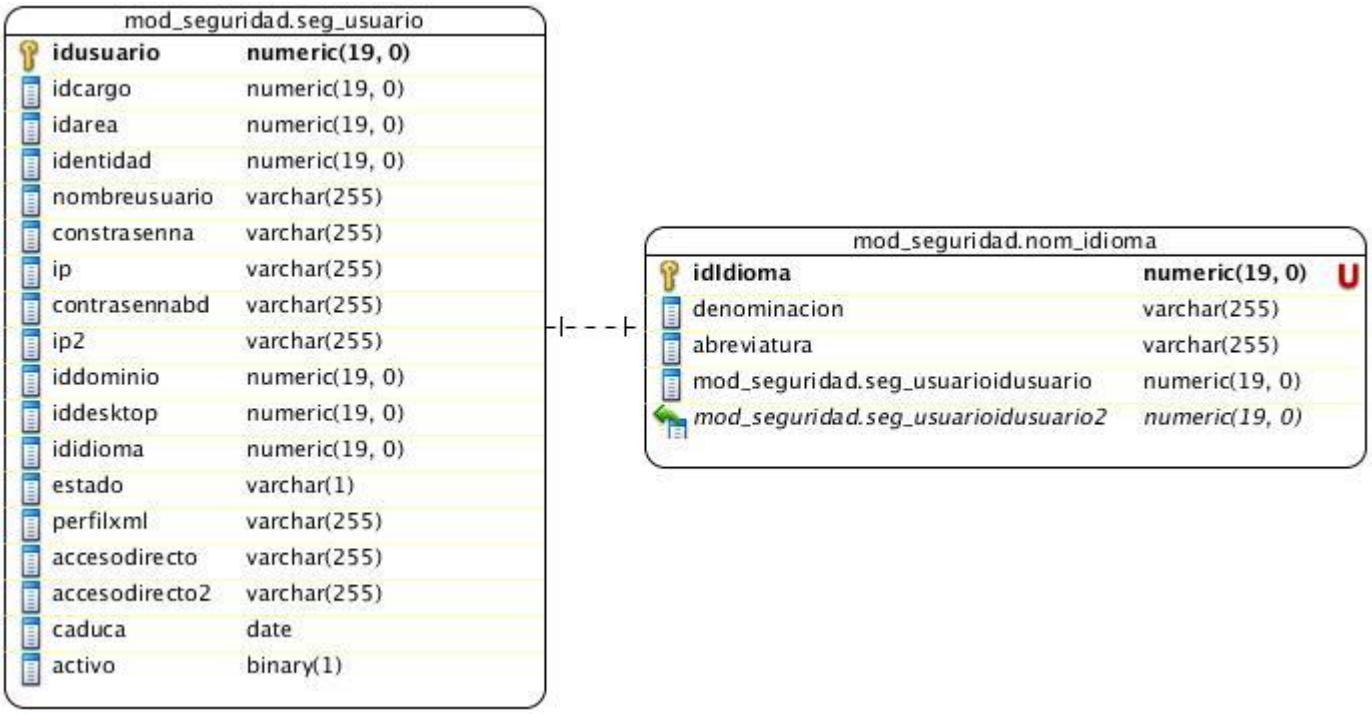


Figura 7. Diagrama Entidad-Relación

En el diagrama representado anteriormente se puede observar la entidad **mod_seguridad.seg_usuario,** la cual permite registrar los datos de los usuarios como idcargo, idarea, identidad, nombreusuario, contrasenna, entre otros. La misma se encuentra dentro del esquema de seguridad y se relaciona con la entidad **mod_seguridad.nom_idioma** que registra la descripción y abreviatura de cada idioma.

2.9 Métricas de Software

La evaluación de un producto, mediante métricas, es un aspecto fundamental a tener en cuenta ya que, las métricas brindan la posibilidad de evaluar la calidad del software a partir de varias reglas definidas claramente. Permiten detectar y corregir los posibles problemas que se puedan presentar durante el proceso de desarrollo y no después de terminado el producto.

La IEEE[2] define las métricas como: *"una medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado*" (((IEEE), 2007).

Según Pressman, la calidad del diseño se puede evaluar aplicando métricas básicas para la calidad del diseño orientado a objetos. Los atributos de calidad involucrados son (Pressman, 2009):

- ✓ **Responsabilidad:** responsabilidad asignada a una clase en un marco de modelado de un dominio o concepto, de la problemática propuesta.
- ✓ **Complejidad de implementación:** grado de complejidad que posee la implementación de un diseño de clases específico.
- ✓ **Reutilización:** grado de reutilización presente en una clase o estructura de clase, dentro de un diseño de software.
- ✓ **Acoplamiento:** grado de dependencia o interconexión de una clase o estructura de clase, con otras, está muy ligada a la característica de reutilización.
- ✓ **Complejidad del mantenimiento:** grado de esfuerzo necesario a realizar para desarrollar un arreglo, una mejora o una rectificación de algún error de un diseño de software. Puede influir indirecta, pero fuertemente en los costes y la planificación del proyecto.
- ✓ **Cantidad de pruebas:** número o grado de esfuerzo para realizar las pruebas de calidad (unidad) del producto (componente, módulo, clase, conjunto de clases, entre otros) diseñado.

Las métricas utilizadas para validar el diseño fueron Tamaño Operacional de Clases (TOC), la cual está dada por el número de métodos asignados a una clase y Relaciones entre Clases (RC), en la cual se tiene en cuenta el número de relaciones de uso de una clase con otra.

2.9.1 Resultados de la aplicación de la métrica TOC al diseño

La figura 8 muestra los resultados obtenidos de la aplicación de la métrica TOC al diseño. Estos resultados se agrupan en los intervalos representados en la gráfica. El gráfico refleja que la mayoría de las clases tienen de uno a diez procedimientos. Esto demuestra que el funcionamiento general del componente está distribuido equitativamente entre las diferentes clases.

[2] **IEEE:** (**I**nstitute of **E**lectrical and **E**lectronics **E**ngineers o Instituto de Ingenieros Eléctricos y Electrónicos)

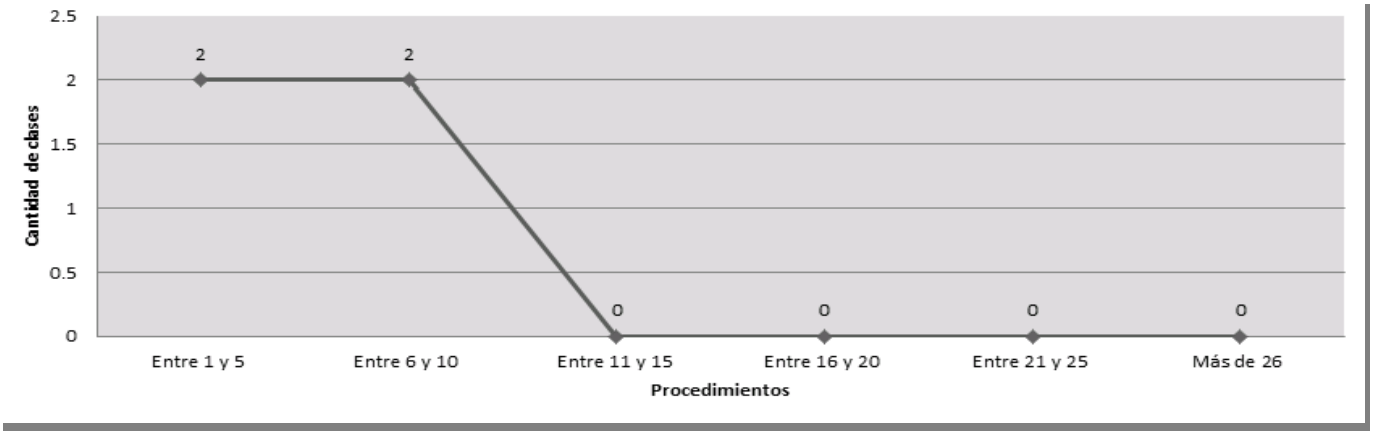


Figura 8.Representación de la evaluación de la métrica TOC

La figura 9 muestra los resultados obtenidos en el instrumento en porciento agrupados en los intervalos definidos.

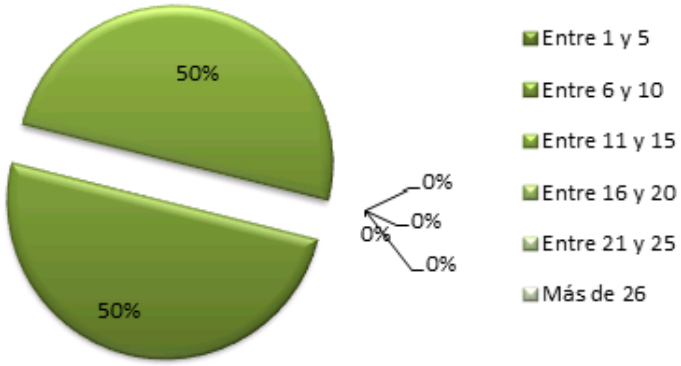


Figura 9.Representación de los resultados obtenidos en la evaluación de la métrica TOC

En la figura 10 se observa la representación de la incidencia de los resultados de la evaluación de la métrica TOC en el atributo Responsabilidad. La gráfica muestra un resultado satisfactorio ya que el 75 % de las clases poseen una baja responsabilidad. Permitiendo que en caso de fallos, como la responsabilidad está distribuida de forma equilibrada ninguna clase sea demasiado crítica como para dejar al sistema fuera de servicio.

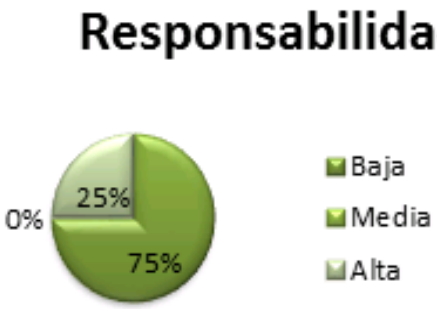


Figura 10.Resultados de la evaluación de la métrica TOC para el atributo responsabilidad

La figura 11 muestra el resultado de la incidencia de la métrica TOC en el atributo Complejidad de Implementación (izquierda). Este gráfico muestra un resultado satisfactorio, pues el 75 % de las clases poseen una baja complejidad de implementación, característica que permite mejorar el mantenimiento y soporte de estas clases. Además la figura muestra en su parte derecha el resultado de la incidencia de la métrica TOC en el atributo Reutilización. Este gráfico muestra que el diseño de la herramienta tiene un grado aceptable pues solo el 22 % del total de las clases posee una baja reutilización.

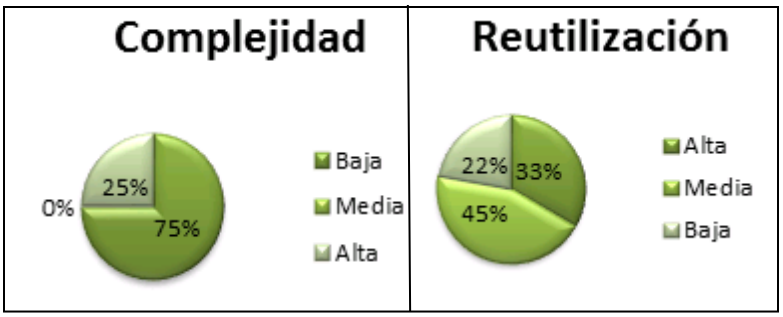


Figura 11.Resultados evaluación de la métrica TOC en: Complejidad de implementación; Reutilización

Teniendo en cuenta los resultados de la aplicación de la métrica TOC al diseño, se puede señalar que en general tiene una calidad aceptable, pues el mayor por ciento del total de las clases posee baja complejidad de implementación y una alta reutilización.

2.9.2 Resultados de la aplicación de la métrica RC

La figura 12 muestra los resultados obtenidos en el instrumento de evaluación de la métrica RC en porcientos agrupados en los intervalos definidos. El gráfico refleja que el 75 % de las clases tienen entre 1 y ninguna dependencia con otra clase, demostrando que casi el total de las clases presentan niveles aceptables de calidad.

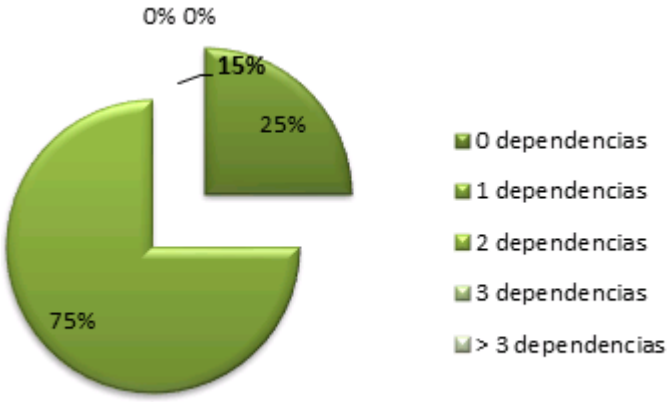


Figura 12.Representación de los resultados obtenidos en los intervalos definidos según la métrica RC

La figura 13 muestra la representación de la incidencia de los resultados de la evaluación de la métrica RC en los atributos: Acoplamiento (izquierda) y Complejidad de mantenimiento (derecha). En el gráfico de la izquierda se evidencia un diseño eficiente al quedar reflejado que existe bajo acoplamiento entre las clases, mientras que el 75 % no presenta dependencias con otras clases, lo que aumenta el grado de reutilización del componente. En el gráfico de la derecha el resultado es favorable ya que el 75 % de las clases presenta baja complejidad de mantenimiento, dándole respaldo a una buena calidad de la arquitectura.

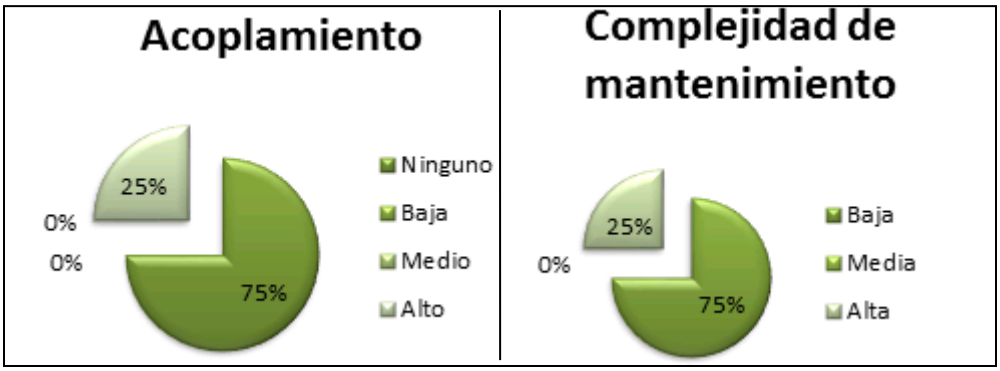


Figura 13.Resultados evaluación de la métrica RC para: Acoplamiento; Complejidad de mantenimiento

La figura 14 muestra la representación de la incidencia de la métrica RC en los atributos: cantidad de pruebas (izquierda) y Reutilización (derecha). El gráfico de la izquierda demuestra que la mayor cantidad de las clases no necesita una cantidad elevada de pruebas según las dependencias entre las clases. Por su parte el gráfico de la derecha muestra que el diseño tiene un 75 % de reutilización de sus clases, constituyendo un factor importante en el desarrollo del software.

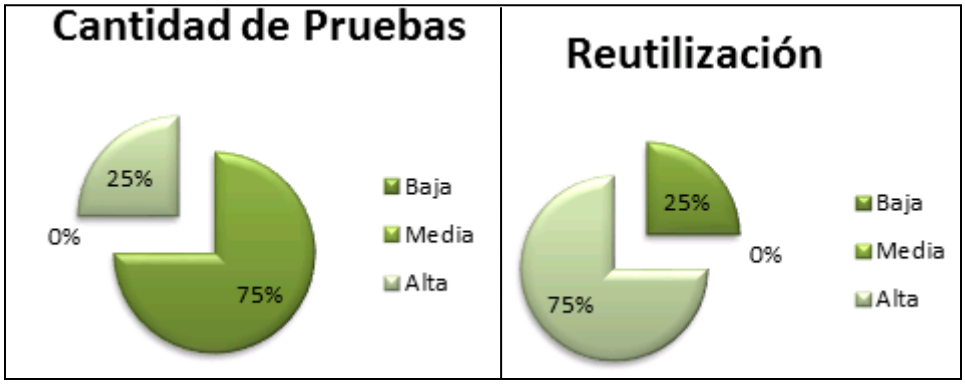


Figura 14. Resultados evaluación de la métrica RC para: Cantidad de pruebas; Reutilización

Analizando los resultados de la evaluación del instrumento de medición de la métrica relación entre clases (RC), se puede concluir que el diseño posee una calidad aceptable, ya que el 75 % de las clases de la solución tienen menos de dos dependencias con otras clases. También se refleja en los resultados el acoplamiento entre clases el cual es mínimo; que la complejidad de mantenimiento así como la cantidad de pruebas son bajas en un 75 %; y que la reutilización se comporta en un valor elevado del 75 %. Este resultado demuestra que los atributos de calidad se encuentran en niveles satisfactorios. Los instrumentos y las tablas de resultados del instrumento de medición de la métrica RC se pueden observar en los productos de trabajo.

Conclusiones del capítulo

En el capítulo concluido se elaboró el modelo conceptual de la solución, donde se muestra la relación que existe entre cada uno de los conceptos identificados, permitiendo tener un mejor dominio del ambiente del proyecto. Se generaron los artefactos de la disciplina Requisitos donde se aplicaron las técnicas de captura de requisitos: entrevistas, prototipo, tormenta de ideas y observación; permitiendo identificar y describir 17 historias de usuario. Se describe el proceso de administración y validación de los requisitos permitiendo tener un mejor dominio de las capacidades o condiciones que el sistema debe cumplir.

Además se realizó el Análisis y Diseño de la propuesta de solución que servirá de base para llevar a cabo el proceso de implementación del componente Internacionalización, permitiendo diseñar mediante el Diagrama de clases del diseño la estructura de las clases y las relaciones que existen entre ellas, quedando reflejado en el diagrama el uso de los patrones de diseño GRASP y GOF.

La evaluación aplicada al diseño mediante las métricas TOC y RC, evidencia niveles satisfactorios como el 75 % de baja responsabilidad y el 45 % de reutilización de las clases del sistema, así como los niveles de los restantes indicadores evaluados.

CAPÍTULO III: IMPLEMENTACIÓN Y PRUEBA

Introducción

El siguiente capítulo aborda los estándares de codificación a utilizar en el componente Internacionalización para los sistemas que se construyen sobre el marco de trabajo Sauxe, así como lo referente a las pruebas realizadas al sistema para validar su correcto funcionamiento. En el desarrollo se describen los estándares de codificación utilizados en función de garantizar un mejor entendimiento y legibilidad del código fuente. Se muestran los casos de pruebas diseñados para realizar las pruebas funcionales, así como los resultados obtenidos, validando que los requisitos identificados fueron implementados correctamente. Además se evidencian los resultados de la evaluación mediante métricas de diseño, pruebas de caja blanca, caja negra y el pre-experimento, métodos escogidos para validar la solución.

3.1 Modelo de implementación

El modelo de implementación se inicia a partir de los resultados obtenidos en el diseño y describe cómo los elementos del modelo de diseño se implementan en términos de componentes. Detalla también cómo se organizan los componentes de acuerdo con los mecanismos de estructuración disponibles en el entorno de implementación y en el lenguaje de programación utilizado. Muestra cómo dependen los componentes unos de otros, además de los recursos necesarios para poder ejecutar la herramienta desarrollada (Acuña, 2013).

3.1.1Diagrama de componentes

Los diagramas de componentes describen los elementos físicos (o componentes) del sistema y sus relaciones, muestran las organizaciones y dependencias lógicas entre componentes de software, sean éstos componentes de código fuente, binarios o ejecutables. El diagrama de componentes de la solución propuesta está representado por los componentes de Idioma, Calendario y Región, que integran el componente de Internacionalización, este último se encuentra dentro del paquete de Seguridad y brinda servicios al componente de Usuario encargado de consumir los servicios de idioma para asociar un usuario con el idioma especificado. El mismo se muestra a continuación:

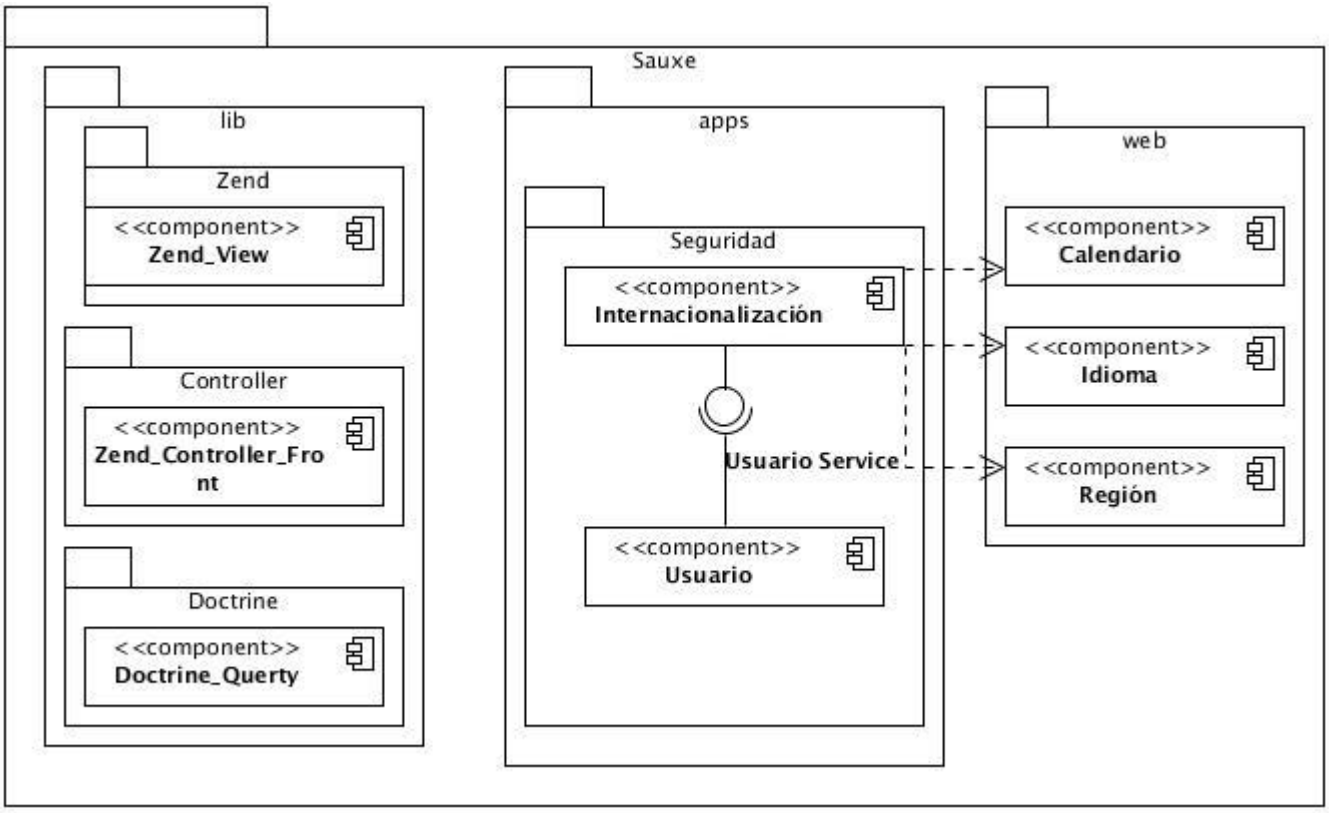


Figura 15.Diagrama de componentes

3.2 Estándares de codificación

Los estándares de codificación ayudan a garantizar que todos los usuarios tengan acceso a la información que el sitio web brinda a sus visitantes, además será más fácil para las personas con necesidades especiales utilizar la web. Por estas razones, la Universidad Nacional Autónoma de México (UNAM) la recomienda que todos los desarrolladores web se apeguen a los estándares de codificación (UNAM, 2015).

Para el desarrollo del componente se utilizarán algunos de los estándares de codificación y normas propuestos como parte de la línea de arquitectura determinada para el desarrollo del ERP[3] Cuba (Cuba, 2008).

[3] **ERP:** por las siglas en inglés **Enterprise Resource Planning.**

Los estándares utilizados en la codificación fueron los siguientes:

- ✓ **Notación Húngara:** define prefijos para cada tipo de datos y según el ámbito de las variables. La idea de esta notación es la de dar mayor información al nombre de la variable, método o función definiendo en ella un prefijo que indique su tipo de dato o ámbito (Sperberg, 2013).
- ✓ **Notación PascalCasing:** se utiliza para escribir el nombre de los identificadores o palabras de un proyecto colocando en mayúscula la primera letra de cada palabra que forme el nombre del componente (Takeyas, 2013).
- ✓ **Notación CamelCasing:** en la notación de camello se escribe la primera letra de la identificación con minúsculas y la inicial de cada una de las palabras concatenadas se escribe con mayúscula (Takeyas, 2013).

3.2.1 Estándares de nomenclatura

Nomenclatura de las clases

Los nombres de las clases comienzan con la primera letra en mayúscula y el resto en minúscula, en caso de que sea un nombre compuesto se empleará notación PascalCasing. Con sólo leerlo se reconoce el propósito de la misma.

Ejemplo: NomIdioma.

Nomenclatura según el tipo de clases

Clase controladora: después del nombre llevan la palabra “Controller”.

Ejemplo: GestnomidiomaController

Nomenclatura de las funciones

El nombre a emplear para las funciones se escribe con la primera palabra en minúscula, en caso de que sea un nombre compuesto se empleará notación CamelCasing. En caso de ser una acción de la clase controladora se debe especificar el nombre de dicha acción en minúscula y seguido el sufijo “Action”.

Ejemplo: insertarnomidiomaAction.

Nomenclatura de las variables

El nombre a emplear para las variables se escribe con la primera palabra en minúscula, en caso de que sea un nombre compuesto se empleará notación CamelCasing. Ejemplo: $idioma.

Normas de comentariado

Se debe comentar todo lo que se haga dentro del desarrollo, establecer las pautas que conlleven a lograr un código más legible y reutilizable y así se pueda aumentar su mantenimiento a lo largo del tiempo. Los

comentarios deben ser lo bastante claros y precisos de forma tal que se entienda el propósito de lo que se está desarrollando.

- ✓ **En las clases**

Antes de la declaración de una clase se escribe una breve descripción donde se explica el propósito de la misma. Dicha descripción puede contener el nombre de la clase, el autor, el paquete al que pertenece, la versión, entre otros datos. Se puede estructurar de la forma que se muestra a continuación:

/**
* Nombre de la clase *
* Descripción *
* @author *
* @package *(módulo)
* @subpackage *(sub módulo)
* @copyright *
* @version (versión - parche)
*/

- ✓ **En las funciones**

Antes de la declaración de la función se escribe una breve descripción donde se explica el propósito de la misma, el autor, los parámetros y, de ser necesarios, algunos señalamientos. Se escribe de la siguiente forma:

/**
* Nombre de la función *
* Descripción *
* @author * (en caso de que no sea el autor de la clase)
* @param *(los parámetros que se le pasan a la función con su descripción)
* @throws *(en caso de que muestre una excepción)
* @return *(se coloca lo que devuelve la función y un comentario)*/
*/

- ✓ **Estilo del código**

En la implementación, al escribir las sentencias en php, se utilizarán los tabs del lenguaje como se muestra en la Figura 17:

```php
<?php
//código
?>
```

Figura 16.Estilo de código

Sangría o indexado

La política de sangría a utilizar en la implementación es por tabulación.

```php
<?php
/**
 * Indentation
 */
class Example {
    var $theInt = 1;
    function foo($a, $b) {
    switch ( $a) {
    case 0 :
    $Other->doFoo ();
    break;
    default :
    $Other->doBaz ();
    }
    }
    function bar($v) {
    for($i = 0; $i < 10; $i ++) {
    $v->add ( $i );
    }
    }
}
?>
```

Figura 17.Estilo del código: Sangría o indexado

Brazas o llaves

En la declaración de clases o interfaces, métodos, bloques y switch, la apertura de llaves se hace en la misma línea. Ejemplo:

```php
<?php
/**
 * Braces
 */
interface EmptyInterface {
}

class Example {
    function bar($p) {
        for($i = 0; $i < 10; $i ++) {
        }
        switch ( $p) {
            case 0 :
                $fField->set ( 0 );
            break;
            case 1 :
                {
                    break;
                }
            default :
                $fField->reset ();
        }
    }
}
?>
```

Figura 18.Estilo del código: brazas o llaves

Espacios en blanco

La declaración de los espacios en blanco en los arreglos se hace tal y como se muestra a continuación.

Ejemplo:

```php
<?php
list ( $a, $b ) = array (1, 2, 3 );
$array = array (1 => 2, 2 => 3 );
$array [$i]->foo ();
$array [] = 'first cell';
?>
```

Figura 19.Estilo de código. Espacio en blanco_Arreglos

3.3 Pruebas de software

La IEEE plantea que “la prueba es la actividad en la cual un sistema o componente es ejecutado bajo condiciones específicas, se observan y almacenan los resultados y se realiza una evaluación de algún aspecto del sistema o componente” (IEEE, 1991).

Pressman por su parte, asegura que además de detectar errores, las pruebas tienen como objetivo medir el grado en que el software cumple con los requisitos. También expone que hay dos enfoques principales, las pruebas de “caja blanca” y las pruebas de “caja negra” (Pressman, 2005).

En la disciplina de pruebas internas que propone la metodología se ejecutaron las pruebas de Caja blanca y Caja negra donde se construyeron los diseños de casos de pruebas como productos de trabajos.

3.3.1 Pruebas estructurales o de Caja blanca

También conocidas como pruebas de “caja de cristal”, es un método de diseño de casos de pruebas que utiliza la estructura de control del diseño procedimental para obtener los casos de pruebas que garanticen que: (Pressman, 2005)

- ✓ Se ejerciten al menos una vez todas las rutas independientes del módulo.
- ✓ Se ejecuten todas las decisiones lógicas en sus opciones verdadera y falsa.
- ✓ Se ejerciten todos los bucles en sus límites.
- ✓ Se usen las estructuras internas de datos para asegurar su validez.

La técnica utilizada para realizar las pruebas de caja blanca del componente Internacionalización es la del **camino básico**. La misma permite obtener una medida de la complejidad lógica de un diseño procedimental y usar esa medida como guía para la definición de un conjunto básico de caminos de ejecución. Para obtener el conjunto de caminos independientes se construye el grafo de flujo asociado y

se calcula su complejidad ciclomática. Los casos de pruebas obtenidos garantizan que se ejecute al menos una vez cada sentencia del programa (Pressman, 2005).

A continuación se muestra el fragmento del código utilizado durante la aplicación de esta técnica a la funcionalidad ***cargarjsonAction()***. El grafo generado se muestra en la figura 22.

```
function cargarjsonAction() {
    $abrevIdioma = $this->_request->getPost('idioma');
    $refFuncionalidad = $this->_request->getPost('referencia');

    $cosas = split(DIRECTORY_SEPARATOR, $refFuncionalidad);

    $dir = '..' . DIRECTORY_SEPARATOR . '..' . DIRECTORY_SEPARATOR . 'apps';

    foreach ($cosas as $key => $value) {
        if ($value == "index.php" || $value == "patdsi.php") {
            $dir .= DIRECTORY_SEPARATOR . "views" . DIRECTORY_SEPARATOR . "idioma" . DIRECTORY_SEPARATOR . $abrevIdioma;
        } else {
            $dir .= DIRECTORY_SEPARATOR . $value;
        }
    }
    $dir .= ".json";

    if (file_exists($dir)) {
        $datos = file_get_contents($dir);
        echo json_encode($datos);
    } else {
        echo"{'codMsg':1,'mensaje': perfil.etiquetas.NoExiste}";
    }

    return;
}
```

*Figura 20.Código fuente de la funcionalidad **cargarjsonAction()***

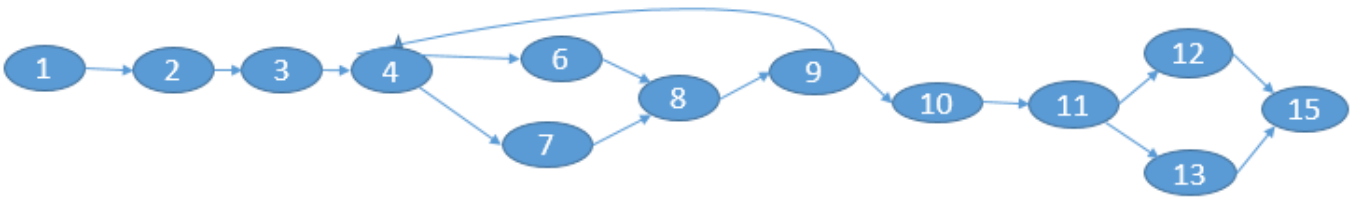


*Figura 21.Grafo de flujo asociado a la funcionalidad **cargarjsonAction ()***

La complejidad ciclomática es la métrica de software que proporciona una medición cuantitativa de la complejidad lógica de un programa. Esta métrica calcula la cantidad de caminos independientes de cada

una de las funcionalidades del programa. También provee el límite superior para el número de pruebas que se deben realizar para asegurar que se ejecute cada sentencia al menos una vez (Pressman, 2005).

Luego de haber construido el grafo se realiza el cálculo de la complejidad ciclomática mediante las tres fórmulas descritas a continuación, las cuales deben arrojar el mismo resultado para asegurar que el cálculo de la complejidad sea el correcto.

1. V (G) = R donde **R** representa la cantidad total de regiones.

V (G) = 4

2. V (G) = A – N + 2 donde **A** es el número de aristas del grafo de flujo y **N** es el número de nodos del mismo.

V (G) = 17 – 15 + 2

V (G) = 4

3. V (G)=P+1 donde **P** es el número de nodos predicado contenidos en el grafo de flujo (se denomina nodo predicado a los nodos de los cuales parten dos o más aristas).

V (G)=3+1

V (G)= 4

Posterior al cálculo de las fórmulas mencionadas, se pudo comprobar que las tres dan el mismo resultado. Por tanto se puede afirmar que la complejidad ciclomática del método cargarjsonAction () tiene un valor de cuatro. Este valor significa que existen cuatro caminos posibles para el flujo del método. Este valor representa el número mínimo de casos de pruebas para el procedimiento tratado.

Camino básico #1: 1 – 2 – 3 – 5 – 6 – 8 – 9 – 10 – 11 – 12 – 14 – 15

Camino básico #2: 1 – 2 – 3 – 5 – 7 – 8 – 9 – 10 – 11 – 12 – 14 – 15

Camino básico #3: 1 – 2 – 3 – 5 – 6 – 8 – 9 – 10 – 11 – 13 – 14 – 15

Camino básico #4: 1 – 2 – 3 – 5 – 7 – 8 – 9 – 10 – 11 – 13 – 14 – 15

Además, este valor representa la mínima cantidad de casos de prueba para el procedimiento, teniendo en cuenta que se realiza un caso de prueba por cada camino básico:

Caso de prueba para el camino básico #1: Si ($value == "index.php" || $value == "patdsi.php") = true.

Caso de prueba para el camino básico #2: Si ($value == "index.php" || $value == "patdsi.php") = false

Caso de prueba para el camino básico #3: Si (file_exists ($dir)) = true.

Caso de prueba para el camino básico #4: Si (file_exists ($dir)) = false.

3.3.2 Pruebas funcionales o de Caja negra

Este tipo de pruebas, también conocidas como “de comportamiento”, se concentran en los requisitos funcionales del software y descubren una clase diferente de errores de los que se descubrirán con los métodos de caja blanca (Pressman, 2005). Las pruebas de caja negra tratan de encontrar errores en las siguientes categorías:

- ✓ Funciones correctas o faltantes.
- ✓ Errores de interfaz.
- ✓ Errores en estructuras de datos.
- ✓ Errores de comportamiento o desempeño.
- ✓ Errores de inicialización y término.

Para realizar este tipo de pruebas, se utilizó la técnica de **Partición equivalente,** la cual divide el dominio de entrada de un programa en clases a partir de las cuales pueden derivarse casos de pruebas, permitiendo examinar los valores válidos e inválidos de estas entradas existentes en el software. Además descubre de forma inmediata una clase de errores que, de otro modo, requerirían la ejecución de muchos casos antes de detectar el error genérico. Esto permite reducir el número de casos de prueba a elaborar (Pressman, 2005).

Durante el desarrollo de las pruebas se elaboraron un total de 13 diseños de casos de pruebas por cada requisito funcional, ver en los productos de trabajo que se entregaron a la analista principal del departamento.

Las no conformidades encontradas en las pruebas internas se clasificaron en significativas y no significativas y se dividen en:

- ✓ No conformidades detectadas en la aplicación.
- ✓ No conformidades detectadas en la documentación.

En la Tabla 3 se muestra la cantidad de no conformidades detectadas en la aplicación y la documentación por cada iteración.

Tabla 3.Resultados de las pruebas internas por cada iteración

No conformidades	Primera Iteración	Significativas	No significativas	Segunda Iteración
Aplicación	9	9	0	0
Documentación	6	1	5	0
Total	15	10	5	0

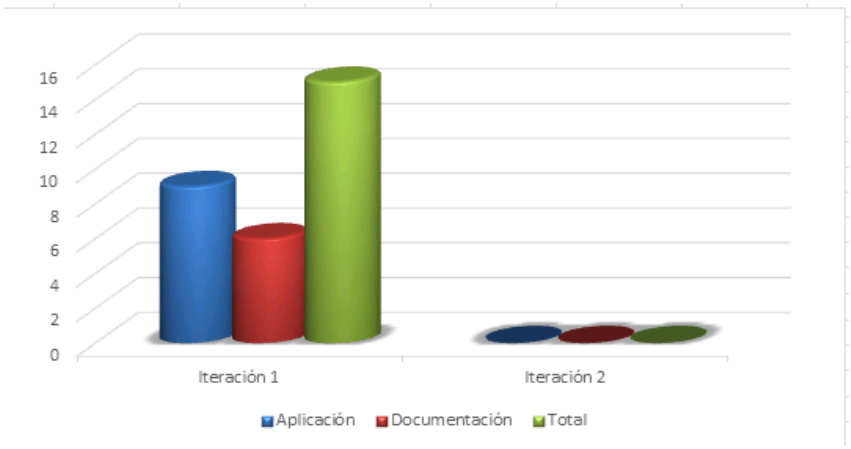


Figura 22.Representación gráfica de los resultados de las pruebas internas por cada iteración

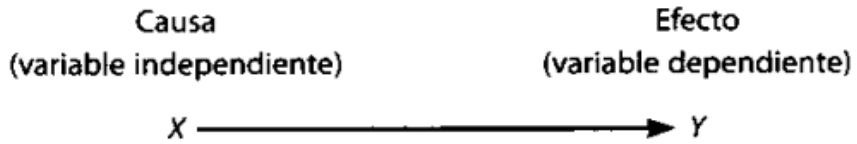
En la disciplina de Pruebas de aceptación se realizó una revisión por parte de los especialistas del departamento de Desarrollo de Componentes, con el objetivo de verificar que el componente cumple con las especificaciones funcionales solicitadas, se detectaron nueve no conformidades en la aplicación, las cuales fueron resueltas para emitir que el componente se encuentra listo para ser usado, ver Anexo 7.

3.4 Validación de la investigación

Luego de definido el problema de la investigación, el alcance inicial y la idea a defender se procede a realizar el diseño que refiere al plan o estrategia que se desarrolla para obtener la información que se requiere en una investigación. Permitirá evaluar que el tiempo de configuración del idioma y las regiones ha disminuido para un valor Y inicial en un valor (1/X) multiplicado por Y utilizando el mismo procedimiento.

Según Sampieri la investigación puede clasificarse en investigación experimental e investigación no experimental. A su vez, la primera puede dividirse de acuerdo con las clásicas categorías de Campbell y Stanley (1966) en: pre-experimentos, experimentos puros y cuasi-experimentos.

La esencia de la concepción de experimento es que requiere la manipulación intencional de una acción para analizar sus posibles resultados, es decir, se manipulan intencionalmente una o más variables independientes (supuestas causas-antecedentes), para analizar las consecuencias que la manipulación tiene sobre una o más variables dependientes (supuestos efectos-consecuentes), como se muestra en el esquema de experimento y variables que propone Sampieri.



En la investigación se propone realizar un diseño experimental porque permite establecer el posible efecto de una causa que se manipula sobre otras variables (las dependientes) en una situación de control. Ejemplo de la relación de la variable independiente y dependiente de la investigación.

Componente Internacionalización _____ disminuye ⟶ Tiempo

El término manipulación es sinónimo de hacer variar o asignar distintos valores a la variable independiente. La variable dependiente no se manipula, sino que se mide para ver el efecto que la manipulación de la variable independiente tiene en ella.

Un requisito importante que todo experimento debe cumplir es el control o la validez interna de la situación experimental, es decir, tener "control" significa saber qué está ocurriendo realmente con la relación entre las variables independientes y las dependientes, mientras que la validez interna es el grado de confianza que se tiene de que los resultados del experimento se interpreten adecuadamente y sean válidos. Cuando el grado de control es mínimo se aplica un pre-experimento como es el caso. Este diseño ofrece una ventaja existe un punto de referencia inicial para ver qué nivel tenía el grupo experimental en la variable dependiente antes de desarrollado el componente Internacionalización (Roberto Hernández Sampieri, 2006).

Aplicación del diseño pre-experimental

1. Se define la variable independiente "Componente Internacionalización" y la dependiente "tiempo de configuración del idioma y las regiones".
2. Se selecciona como instrumento para medir la variable dependiente la encuesta, como se muestra en el Anexo 4.
3. Se selecciona una muestra aleatoria de personas para el experimento, grupo experimental compuesto por seis especialistas, de los cuales cincos son desarrolladores y uno es analista, dos trabajadores son del departamento de Desarrollo de Componentes y cincos son del proyecto Sistema de Gestión Comercial de Importación, la mayoría tienen más de dos años desempeñando el rol y trabajando con el marco de trabajo Sauxe.
4. Se aplica la encuesta y se procede a realizar una evaluación de los resultados.

Evaluación de los resultados de la encuesta:

Para la evaluación de los resultados de la encuesta se definieron los siguientes indicadores, mostrándose los resultados en las siguientes tablas.

Escala: R: Relevante PR: Poco relevante NR. No relevante

Encuestados

1. Eddie N. Beltrán González representa el encuestado A
2. Sonia Fernández Henríquez representa el encuestado B
3. Yuniel Martínez Ordaz representa el encuestado C
4. Yoenry Venega Hechavarría representa el encuestado D

Departamento: Desarrollo de componentes.

1. Katia Saria Preval representa el encuestado E
2. Claudia Bravo Batista representa el encuestado F

Indicador 1. El grado de relevancia de los elementos de internacionalización propuestos.

Tabla 4. Indicador 1

Internacionalización	**Encuestados**						**Resultado**
	A	B	C	D	E	F	
Idioma	R	R	R	R	R	R	100 % R
Formatos de fecha y Hora	R	R	R	R	R	R	100% R
Zona Horaria	R	R	PR	R	R	R	0.83% R

Indicador 2. El tiempo empleado para la configuración de los ficheros del idioma de un componente, antes y después de construido el componente Internacionalización.

Para evaluar este indicador se realizó la configuración del fichero de idioma español, en el componente Multimoneda.

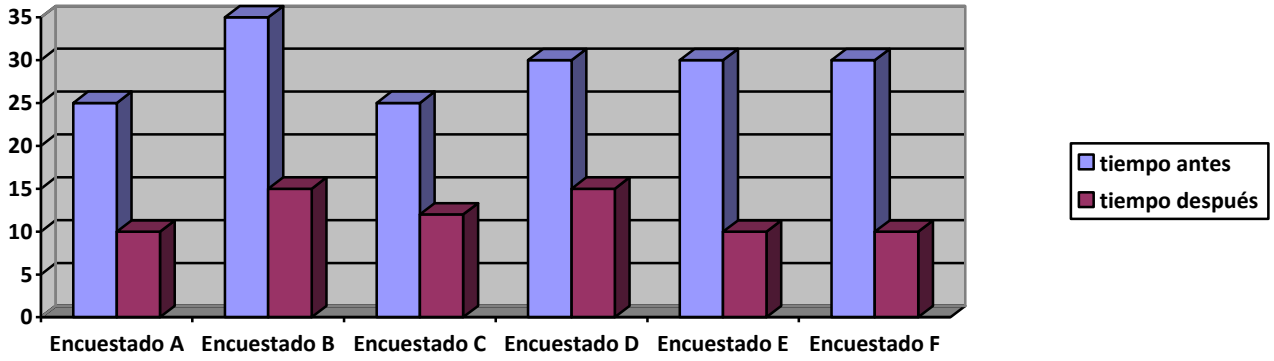


Figura 23 Tiempo de configuración del idioma español en el componente Multimoneda

Tiempo antes: el tiempo (minutos) que demoró la configuración del fichero de idioma de forma manual.

Tiempo después: el tiempo (minutos) que demoró la configuración del fichero de idioma con el componente Internacionalización.

Indicador 3. El grado de relevancia que tiene la integración del componente Internacionalización en los productos que se construyen con el marco de trabajo Sauxe.

Tabla 5. Indicador 3

Componente Internacionalización	**Encuestados**						**Resultado**
	A	B	C	D	E	F	
	R	R	R	R	R	R	100 % R

Indicador 4. El grado de satisfacción de los especialistas en realizar la configuración con el componente Internacionalización. Es importante destacar que este indicador fue propuesto por Eddie N. Beltrán González, desarrollador del proyecto de Sistema de Gestión de Importación y Exportación (BK Import/Export).

Escala: E: Excelente B: Bueno R: Regular M: Malo

El grado de satisfacción de los especialistas en realizar la configuración utilizando el componente Internacionalización.

Tabla 6. Indicador 4

Componente Internacionalización	**Encuestados**						**Resultado**
	A	B	C	D	E	F	
	B	B	B	B	B	B	100 % B

El grado de satisfacción de los especialistas en realizar la configuración de forma manual

Tabla 7. Indicador 4

Sin el componente Internacionalización	**Encuestados**						**Resultado**
	A	B	C	D	E	F	
	M	M	M	M	M	M	100 % M

Conclusiones del capítulo

El Modelo de componentes permitió desplegar la estructura física del componente Internacionalización y las relaciones existentes con los restantes componentes del marco de trabajo Sauxe.

Las normas y estándares de codificación utilizados permitieron brindarle legibilidad y escalabilidad al código fuente.

Se realizó una evaluación del correcto funcionamiento del sistema, identificando los errores en las interfaces y en código fuente de la aplicación. Además, se realizaron las revisiones internas y de aceptación por parte del equipo técnico del departamento donde se atribuye la solución propuesta.

Se validó la investigación realizando un diseño pre-experimental, teniendo en cuenta que el grado de control de la variable independiente era mínimo, que se conocía el estado actual del tiempo de configuración del idioma y las regiones a través de las encuestas realizadas a los especialistas; se pudo observar una aproximación de un estado esperado en la realización de la post prueba; lo que permitió demostrar que el tiempo de configuración del idioma para un componente del marco de trabajo Sauxe, disminuyó en más de un 50% y el grado de satisfacción del desarrollador en el estado esperado era bueno en comparación con el estado actual.

CONCLUSIONES GENERALES

El desarrollo de la investigación permitió arribar a las siguientes conclusiones:

Se realizó un estudio de los conceptos fundamentales asociados al dominio del problema y de los marcos de trabajo que atribuían elementos asociados a la internacionalización, permitiendo tener un mejor dominio de la situación problemática en cuestión. Se definió la metodología que rigió las disciplinas y las actividades a desarrollar durante el proceso de diseño y construcción del componente Internacionalización; así como las tecnologías y herramientas que permitieron construir la solución propuesta.

La captura, especificación y validación de los 18 requisitos funcionales mostrados en cinco agrupaciones de requisitos generales, aportó una explicación escrita del problema y una mayor comprensión del comportamiento que se necesita implementar en la solución

Se diseñó la propuesta de solución permitiendo representar las clases fundamentales del código fuente de la aplicación y las relaciones entre las clases, así como la estructura del componente Internacionalización y las relaciones con los restantes componentes del marco de trabajo.

Se realizó una evaluación del correcto funcionamiento del sistema, identificando los errores en las interfaces y en código fuente de la aplicación.

Se validó la investigación realizando un diseño pre-experimental, lo que permitió demostrar que el tiempo de configuración del idioma para un componente, disminuyó en más de un 50% y el grado de satisfacción del desarrollador era bueno en comparación con el estado actual.

RECOMENDACIONES

Se recomienda que el componente Internacionalización pueda ser utilizado por los productos que utilizan el marco de trabajo Sauxe como base tecnológica con proyección a un mercado internacional.

BIBLIOGRAFÍA

Herramienta para la personalización de los temas de la capa de presentación en el marco de trabajo. . Guevara, William Amed Tamayo. 2014. La Habana: s.n., 2014.

2.0.65, Apache httpd. 2013. The Apache HTTP server Project. http://httpd.apache.org/. [En línea] 9 de 7 de 2013. [Citado el: 8 de 2 de 2015.] http://httpd.apache.org/.

Andrea del Campo. 2014. e.pages. [En línea] 18 de 07 de 2014. http://blog.epages.com.

Apache. 2014. Apache Subversion. [En línea] 2014. [Citado el: 27 de enero de 2014.] http://subversion.apache.org.

Aurora Aparicio, Wilson Daniel Palacios, Ana María Martínez, Isabel Ángel, Cecilia Verduzco, Elisa Retana. 2009. El cuestionario.Métodos de Investigación Avanzada. 2009.

Baryolo, Oiner Gomez. 2010. Solución informática de autorización en entornos multientidad y multisistema. 2010.

BERKANE, Rachid. 2013. unixmen. unixmen. [En línea] 13 de marzo de 2013. [Citado el: 25 de 1 de 2015.] http://aromatix.fr/?p=611.

Cobas, P. M. (2013). Procedimiento para desarrollar la interoperabilidad en proyectos de desarrollo de Sistemas de Gestión Empresarial en Cuba. (2013).

Componentes, Departamento de. 2014. Guía de implementación con ExtJs 4.2 aplicando la arquitectura MVC en la Capa de Presentación del Marco de Trabajo Sauxe. 2014.

Councill, W. T. y Heineman, G. T. 2001. Definition of a software component and its elements. 2001.

Diccionario Ilustrado Océano de la Lengua Española. España: s.n.

django. 2014. django. django. [En línea] 2014. [Citado el: 1 de 3 de 2015.] https://docs.djangoproject.com/en/1.7/topics/i18n/.

djangoproject. 2014. django. django. [En línea] 2014. [Citado el: 1 de 3 de 2015.] https://www.djangoproject.com/.

Eguiluz, Javier. 2014. Desarrollo web ágil con Symfony2. Habana: s.n., 2014.

Esser, S. 2009. Secure Programming with the Zend-Framework. 2009.

-framework-php-yii. 2013. tecnologias-web-traduccion. Tecnologias-web-traduccion. [En línea] 02 de 2013. [Citado el: 19 de 01 de 2015.] http://tecnologias-web-traduccion.blogspot.com/2013/02/gestion-plurales-framework-php-yii.html.

Fry, R. Lommel and D. 2003 “The Localization Industry Primer.”. LISA: Localization Industry Standards Association, 2003.

BIBLIOGRAFÍA

Ganesh, Gunda Sai. 2008. Requirements Engineering: Elicitation Techniques. . Suecia: Universidad del Este Departamento de Tecnología, Matemática y Ciencia de la Computación: 2008. PR003.

González, Mariano. 2012. Herramientas TIC para la internacionalización. Asturia: s.n., 2012.

guiadehostingweb. 2012. www.guiadehostingweb.com. www.guiadehostingweb.com. [En línea] 26 de 9 de 2012. [Citado el: 5 de 2 de 2015.] http://www.guiadehostingweb.com/los-mejores-marcos-de-trabajo-para-php/.

Helmut. 2013. lingohub. blog.lingohub.com. [En línea] 23 de July de 2013. [Citado el: 4 de 3 de 2015.] http://blog.lingohub.com/2013/07/internationalization-how-to-5-most-popular-php-frameworks/.

http://www.codeigniter.com. http://www.codeigniter.com. [En línea] [Citado el: 10 de febrero de 2015.]

Informáticas, U. d. 2014. Metodología de desarrollo para la Actividad productiva de la UCI. La Habana: s.n., 2014.

K. Prada Nicot, H. 2009. Desarrollo de los componentes Puesto de Trabajo y Pagos Adicionales del subsistema Capital Humano integrado al sistema integral de gestión CEDRUX. . 2009.

kohana. kohanaframework.org. kohanaframework.org. [En línea] [Citado el: 28 de 1 de 2015.] https://kohanaframework.org/.

leninmhs. 2013. leninmhs. leninmhs. [En línea] 09 de 04 de 2013. [Citado el: 28 de 02 de 2015.] https://leninmhs.wordpress.com/2013/04/09/entendiendo-yii/.

León, y. o. 2002. 2002.

2013. Lingobit Localizer. [En línea] 2013. http://www.lingobit.com/es/.

M. A. Friedhelm Betz, N. L. 2012. Manual de PHP. 2012.

Manson, Lic. Marcelo. 2014. http://www.monografias.com. evoinfosoc. [En línea] 13 de mayo de 2014. [Citado el: 13 de marzo de 2015.] http://www.monografias.com/trabajos/evoinfosoc/evoinfosoc.shtml.

Moreno, Miguel E. Torres. 2011. Recomendaciones para desarrollar un software internacionalizado. Colombia: Sede Bogotá, 2011.

Pérez Mariñán, P. 2007. Patrones de Diseño. 2007.

PostgreSQ. 2008. Manual del usuario de PostgreSQL. . 2008.

Puro, PGADMIN. Excelente gestor PostGreSQL. 2011. Excelente gestor PostGreSQL. Puro Software. 2011.

R. Frederick, S. 2008. Learning ExtJS. . . 2008.

RapidSVN. 2013. RapidSVN. [En línea] 2013. http://www.rapidsvn.org.

Revista Avances en Sistemas e Informática. 2007. 2007.

BIBLIOGRAFÍA

Ricardo Payán Gómez, Julian Barbosa,Miguel Torres. 2011. Recomendaciones para desarrollar software internacionalizado. Bogota,Colombia : s.n., 2011.

Server, Apache httpd 2.0.65 Released. The Apache HTTP. 2013. The Apache HTTP server Project. Apache httpd 2.0.65 Released. [En línea] 9 de julio de 2013. [Citado el: 8 de febrero de 2015.] http://httpd.apache.org/.

Software, Máster de ingeniería de. 2014. Manual técnicas para la captura de requisitos. 2014.

Sommerville, Ian. 2006. Software Engineering. s.l.: 9th ed. Pearson Education, Inc, 2006.

—. 2006. Software Engineering. s.l.: Pearson Education, Inc., 2006. 9th ed.

Skvorc, Bruno. 2015. sitepoint. sitepoint. [En línea] 28 de 03 de 2015. [Citado el: 20 de 05 de 2015.] http://www.sitepoint.com/best-php-framework-2015-sitepoint-survey-results/.

symfony. 2013. symfony.com. symfony.com. [En línea] 2013. [Citado el: 25 de 2 de 2015.] http://symfony.com/doc/current/book/translation.html.

Technology, British Columbia Institute of. 2014. codeigniter. www.codeigniter.com. [En línea] Copyright 2014 - 2015, 05 de Feb de 2014. [Citado el: 3 de 3 de 2015.] http://www.codeigniter.com/userguide3/general/welcome.html.

Tutorial, Kohana 3 i18n. 2010. blog.mixu.net. blog.mixu.net. [En línea] 11 de noviembre de 2010. [Citado el: 1 de 3 de 2015.] http://blog.mixu.net/2010/11/11/kohana-3-i18n-tutorial/.

Valdés, Damián Pérez. 2007. maestrosdelweb.com. maestrosdelweb. [En línea] 31 de julio de 2007. [Citado el: 9 de 2 de 2015.] http://www.maestrosdelweb.com/los-frameworks-de-php-agilizan-tu-trabajo/.

yiiframework. 2013. www.yiiframework.com. www.yiiframework.com. [En línea] 2013. [Citado el: 17 de 2 de 2015.] http://www.yiiframework.com/doc/guide/1.1/es/topics.i18n.

Roberto Hernández Sampieri, Carlos Fernández Coello, Pilar Baptista Lucio. 2006. Metodología de la Investigación. Cuarta Edición. México: s.n., 2006. ISBN 970-10-5753-8.

ANEXOS

Anexo 1 Entrevista realizada a los especialistas que configuraron los idiomas español e inglés en el marco de trabajo Sauxe.

Entrevistados:

Ing. Mileidy M. Sarduy Pérez.

Graduado de Ingeniería en Ciencias Informáticas en julio del 2009. En el momento de la entrevista se desempeñaba como Jefa del proyecto Gestión Integral de Seguridad (Acaxia).

Ing. Dausbel Torreblanca Pavó

Graduado de Ingeniería en Ciencias Informáticas en julio del 2012. En el momento de la entrevista se desempeñaba como Administrador de la configuración.

Preguntas realizadas durante la entrevista:

¿Cuáles fueron los idiomas que se configuraron en el marco de trabajo Sauxe?

¿Cuál fue el tiempo que demoró realizar la configuración de los idiomas?

¿Cuáles son los elementos que se tuvieron en cuenta para realizar la configuración de los idiomas?

¿Qué recursos se necesitaron para realizar la configuración de los idiomas?

Anexo 2 Encuesta inicial para la internacionalización del marco de trabajo

Encuesta

Componente Internacionalización

La internacionalización es el proceso de diseñar un software que pueda adaptarse a diferentes idiomas y regiones. Actualmente en el marco de trabajo Sauxe, se puede realizar la configuración del idioma permitiendo al usuario que se encuentra logueado en el sistema, acceder por el idioma registrado.

Considera importante la configuración de los diferentes idiomas y regiones en el marco de trabajo Sauxe.

_____ Si _____No. ¿Por qué?

_____

Si usted ha realizado la configuración del idioma en el marco de trabajo Sauxe, describa a su juicio los elementos que tuvo en cuenta para realizar la configuración.

1. Mencione la cantidad de idioma y el tiempo que se empleó para realizar el proceso de configuración del idioma.

_____

_____

2. Mencione los recursos que se necesitaron para realizar la configuración.

a. Cantidad de puestos de trabajo _____

b. Cantidad de personal _____

c. Cantidad de especialistas con conocimientos sobre la configuración del idioma en Sauxe:

_____

d.Otros_____

_____

Las respuestas de esta encuesta servirán de ayuda a los estudiantes del Departamento de Desarrollo de Componentes, durante la investigación del trabajo de diploma.

Muchas gracias por su colaboración.

Anexo 3 Diagramas de clases del diseño

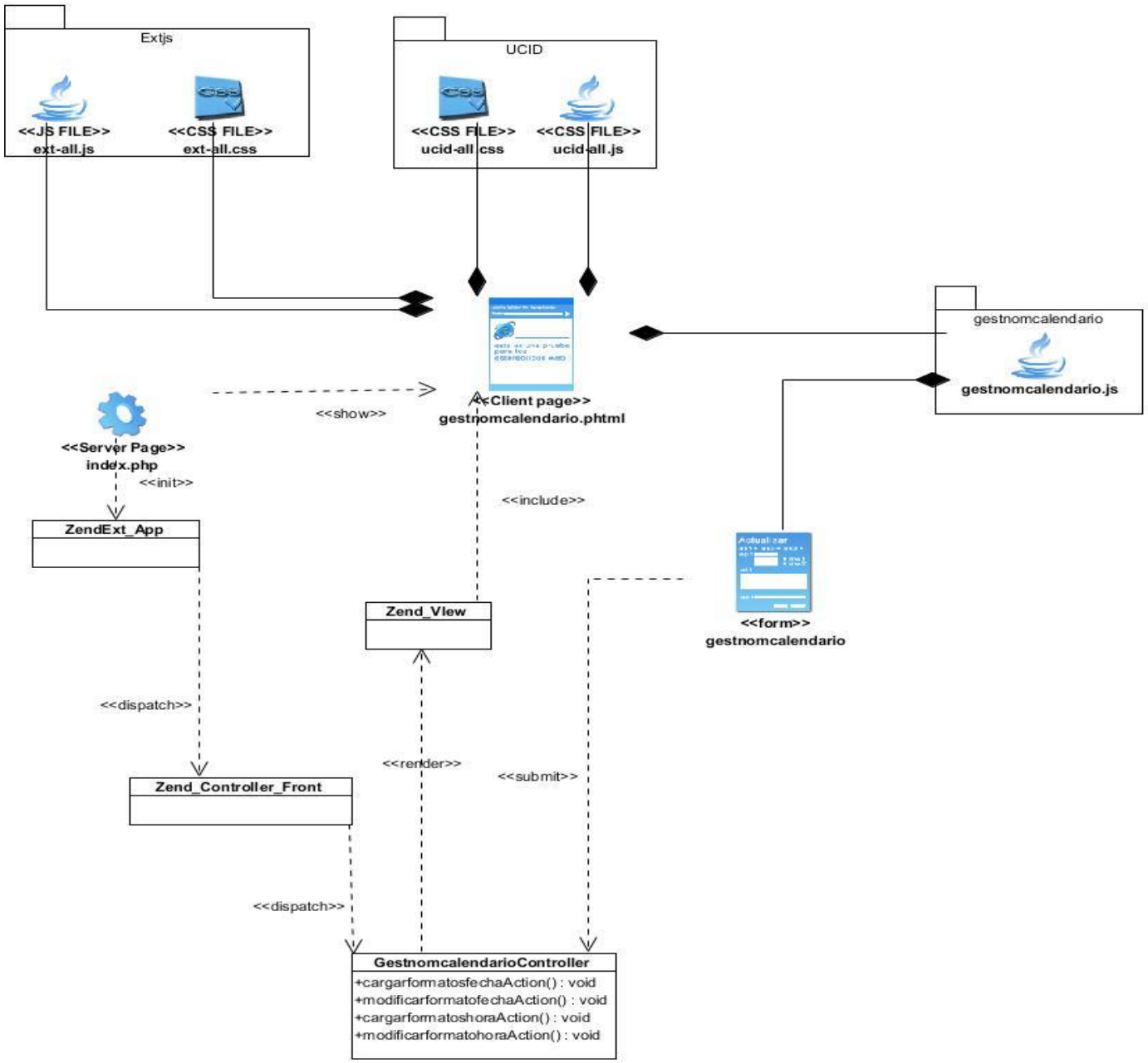


Figura 24. Diagrama de Clases del Diseño del requisito Gestionar calendario

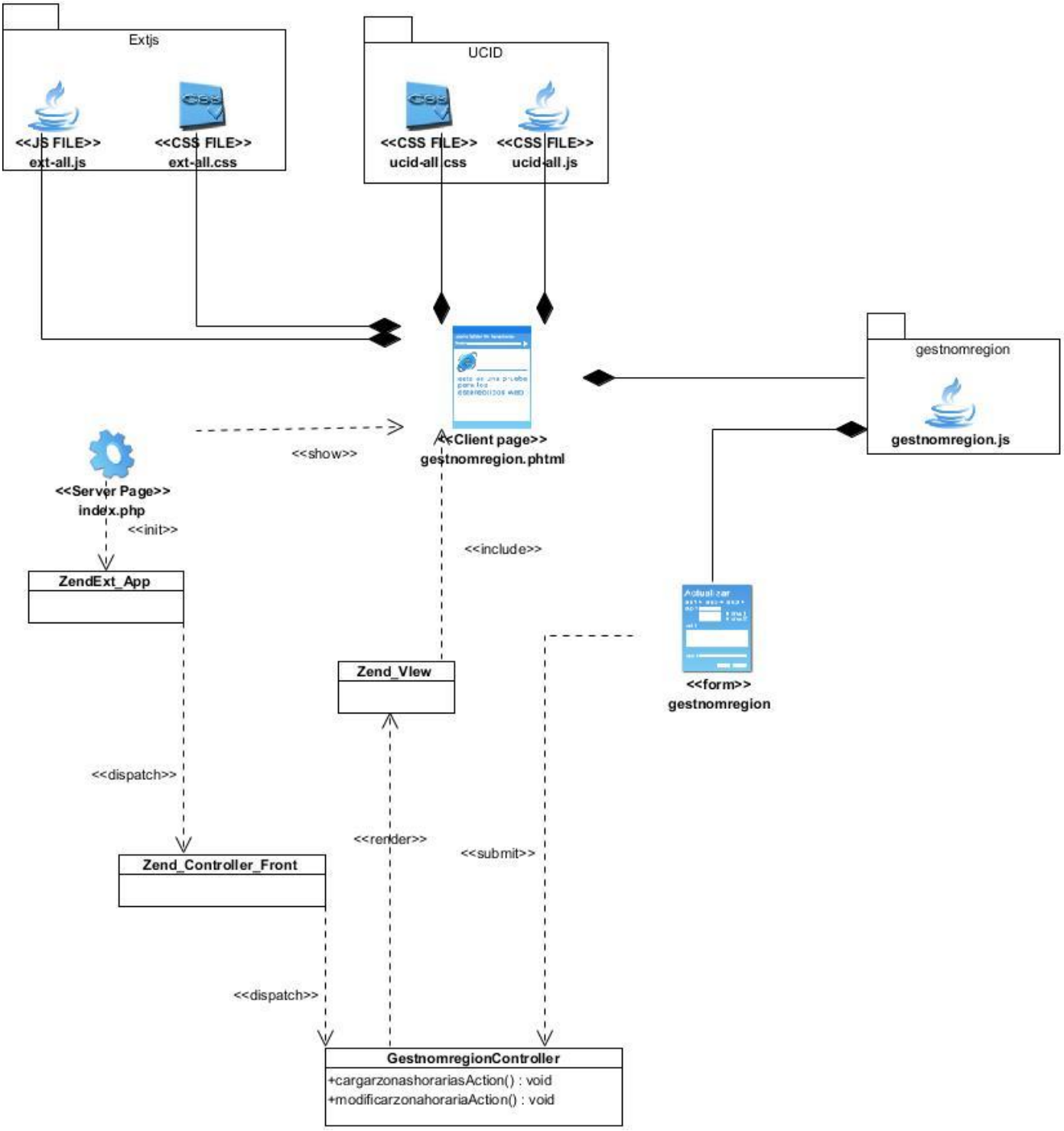


Figura 25. Diagrama de Clases del Diseño del requisito Gestionar región

Anexo 4 Encuesta para la validación del diseño pre-experimental propuesto

Encuestados

Proyecto: Sistema de Gestión Comercial de Importación.

5. Eddie N. Beltrán González
6. Sonia Fernández Henríquez
7. Yuniel Martínez Ordaz
8. Yoenry Venega Hechavarría

Departamento: Desarrollo de componentes.

3. Katia Saria Preval
4. Claudia Bravo Batista

ENCUESTA

Nombre y apellidos: _____

Institución a la que pertenece: _____

Cargo actual: _____

Calificación profesional, grado científico o académico:

Profesor: _____.

Licenciado: _____.

Especialista: _____.

Máster: _____.

Doctor: _____.

Años de experiencia en el cargo: _____.

Con el propósito de validar la investigación propuesta del trabajo de diploma Componente Internacionalización para los sistemas que se construyen sobre el marco de trabajo Sauxe, dirigido a disminuir el tiempo de configuración del idioma y las regiones de las aplicaciones que se construyen sobre el marco de trabajo Sauxe, se propone la siguiente encuesta donde se desea conocer su valoración crítica referente a:

El componente Internacionalización permite configurar el idioma en los productos que utilizan el marco de trabajo Sauxe como base tecnológica.

El tiempo de configuración de los ficheros del idioma de los productos que se construyen con el marco de trabajo Sauxe ha disminuido con la utilización del componente Internacionalización.

El componente Internacionalización permite configurar los formatos de hora y fecha.

El componente Internacionalización permite configurar la zona horaria, en dependencia de la región donde se utilice el producto desarrollado con el marco de trabajo Sauxe.

Indicaciones

Marque con una X la respuesta que usted considera correcta.

1. Especifique el grado de relevancia que usted considera que tiene la configuración de los siguientes elementos de internacionalización propuestos.

   Escala:

   R: Relevante PR: Poco relevante NR. No relevante

Internacionalización	**R**	**PR**	**NR**
Idioma			
Formatos de hora y fecha			
Zona horaria			

2. Especifique aproximadamente el tiempo empleado para realizar la configuración de los ficheros del idioma de un componente antes de utilizar el componente Internacionalización.

   _____ Menos de 10 minutos
   _____ Más de 10 minutos y menos de 30 Minutos.
   _____ Más de 30 minutos

3. Especifique aproximadamente el tiempo empleado para realizar la configuración de los ficheros del idioma de un componente con la utilización del componente Internacionalización.
   _____ Menos de 10 minutos
   _____ Más de 10 minutos y menos de 30 Minutos.
   _____ Más de 30 minutos
4. Especifique el grado de relevancia que tiene la integración del componente Internacionalización en los productos que se construyen con el marco de trabajo Sauxe.
   R: Relevante _____ PR: Poco relevante _____ NR: No relevante _____

Anexo 5 Integración del componente Internacionalización en el sistema Xedro - Apside

Imágenes de la integración del componente Internacionalización en el producto BK-Import, que utiliza como base tecnológica el marco de trabajo Sauxe**.**

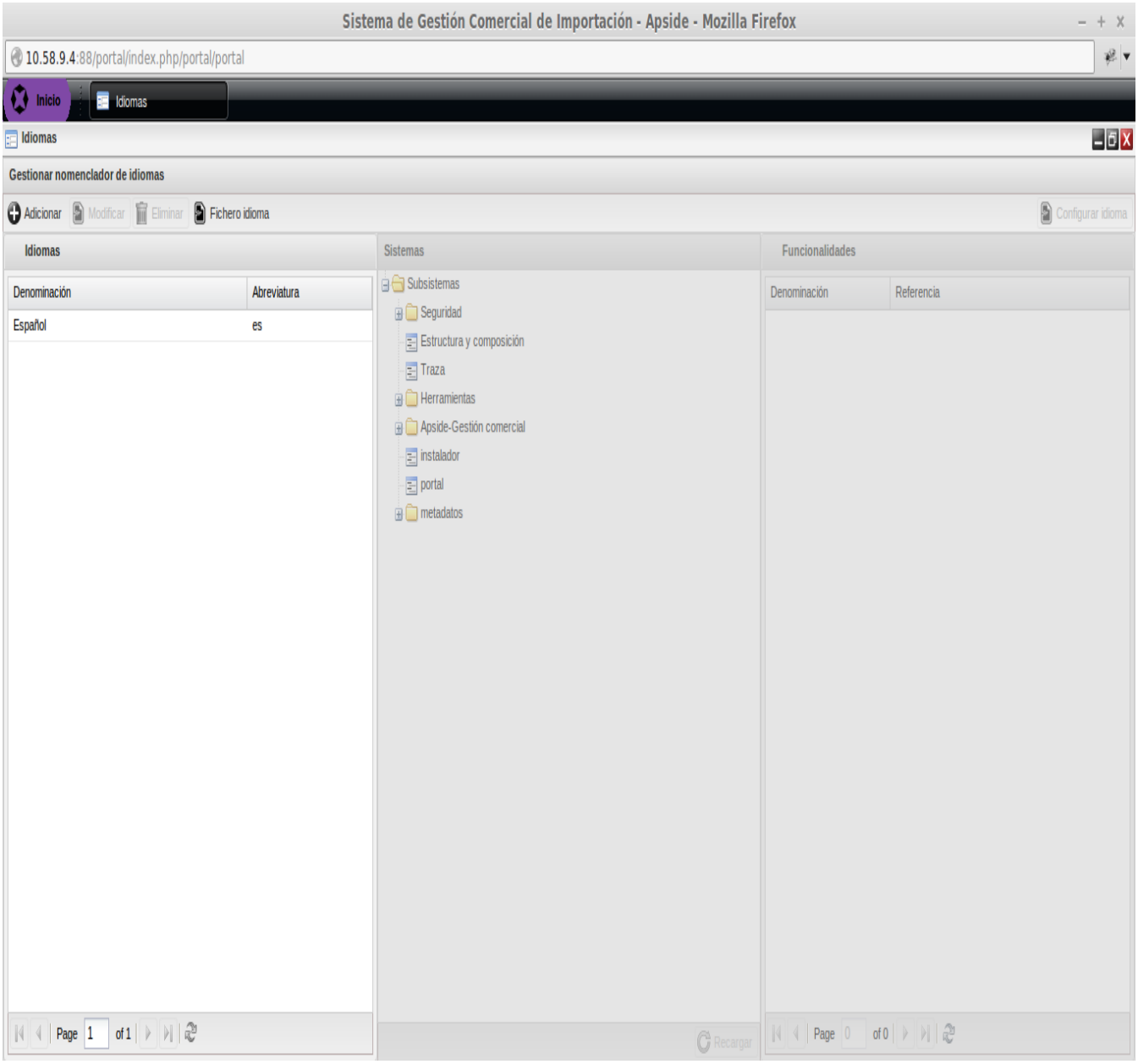


Figura 26.Integración del componente Internacionalización en el sistema Xedro - Apside

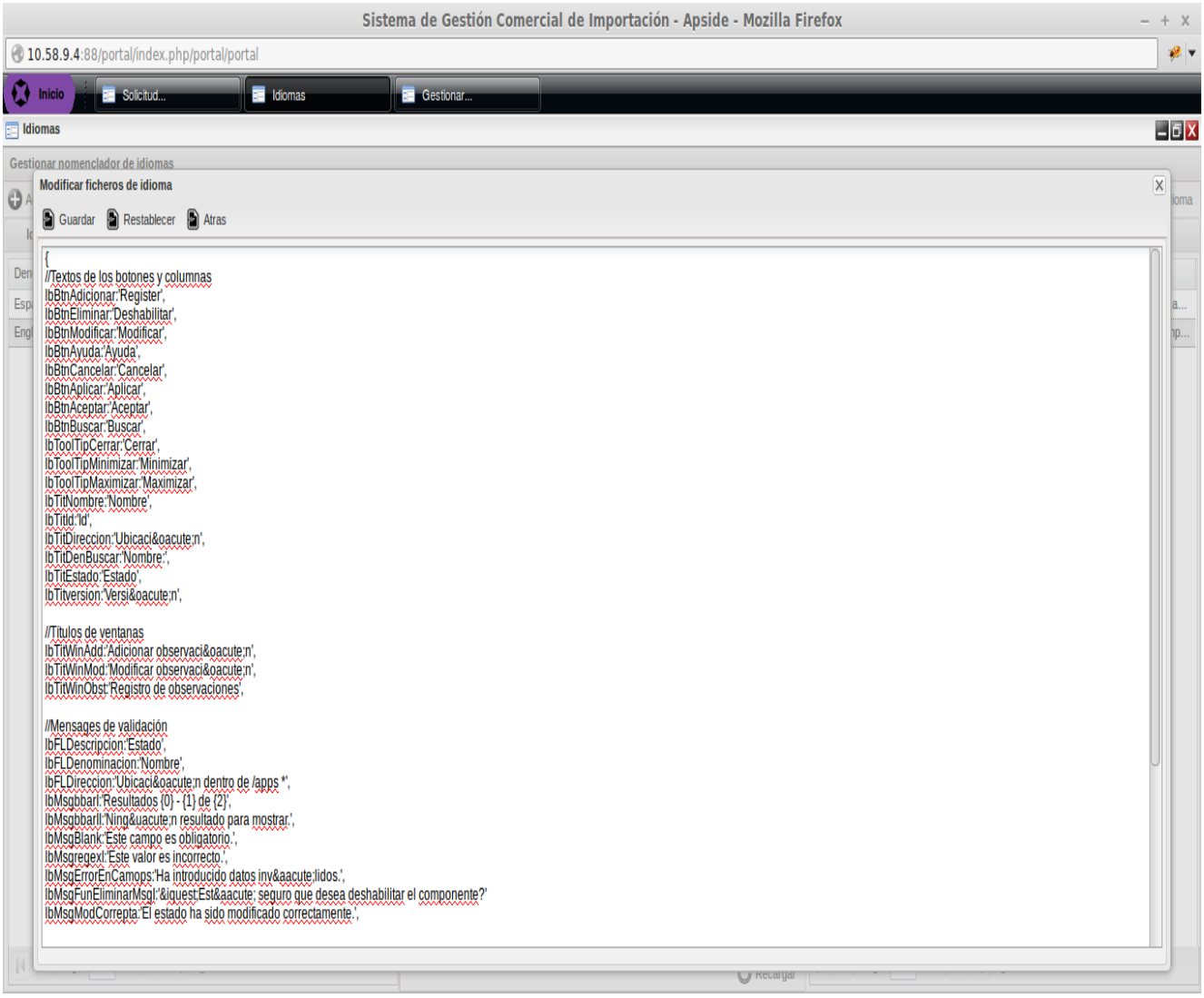
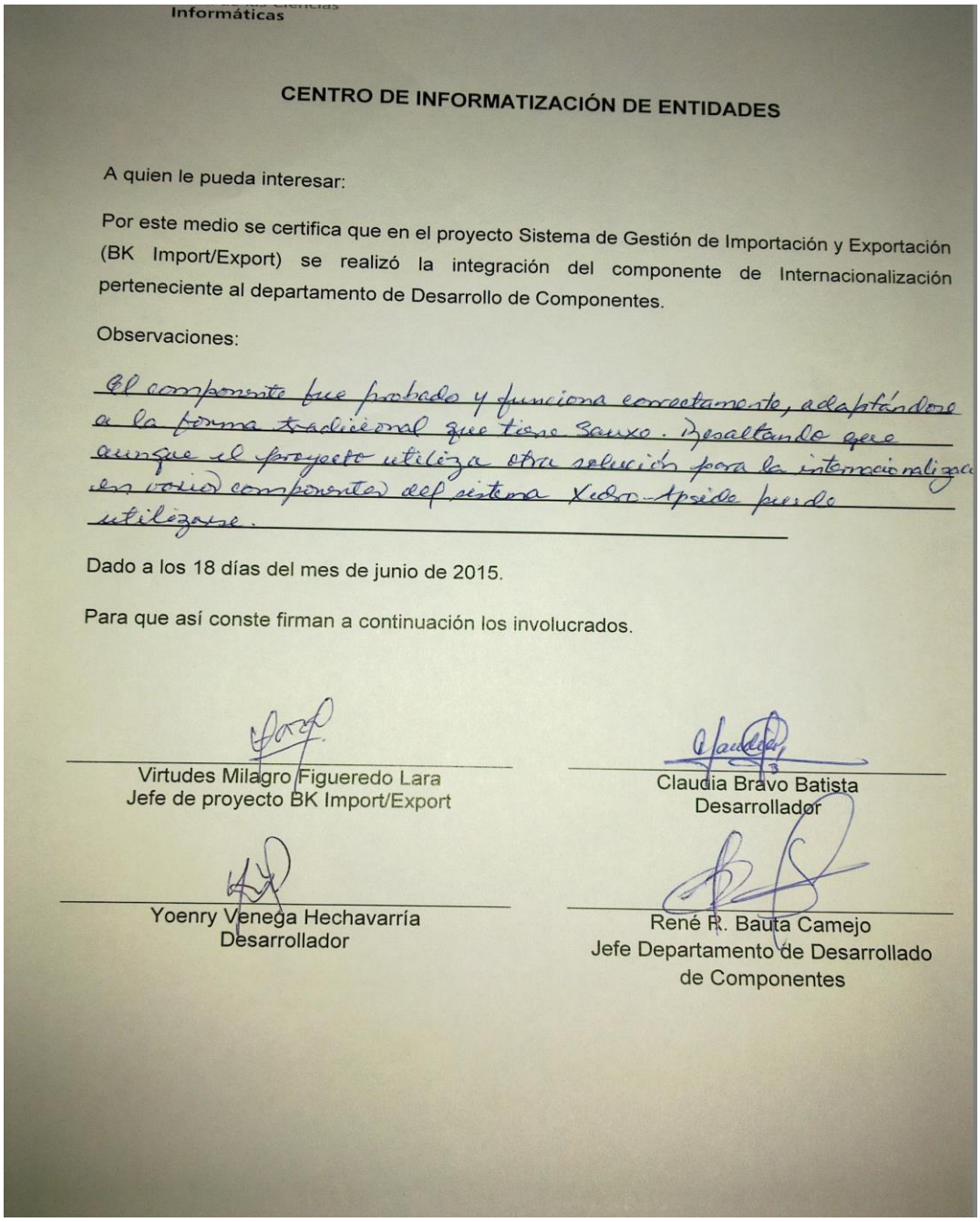


Figura 27. Integración del componente Internacionalización en el sistema Xedro - Apside

Anexo 6 Acta que certifica que el componente Internacionalización fue integrado al sistema Xedro - Apside



Informáticas

CENTRO DE INFORMATIZACIÓN DE ENTIDADES

A quien le pueda interesar:

Por este medio se certifica que en el proyecto Sistema de Gestión de Importación y Exportación (BK Import/Export) se realizó la integración del componente de Internacionalización perteneciente al departamento de Desarrollo de Componentes.

Observaciones:

El componente fue probado y funciona correctamente, adaptándose a la forma tradicional que tiene Xauxo. Resaltando que aunque el proyecto utiliza otra solución para la internacionalización (vario componentes del sistema Xedro-Apside pueden utilizarse.

Dado a los 18 días del mes de junio de 2015.

Para que así conste firman a continuación los involucrados.

Virtudes Milagro Figueredo Lara
Jefe de proyecto BK Import/Export

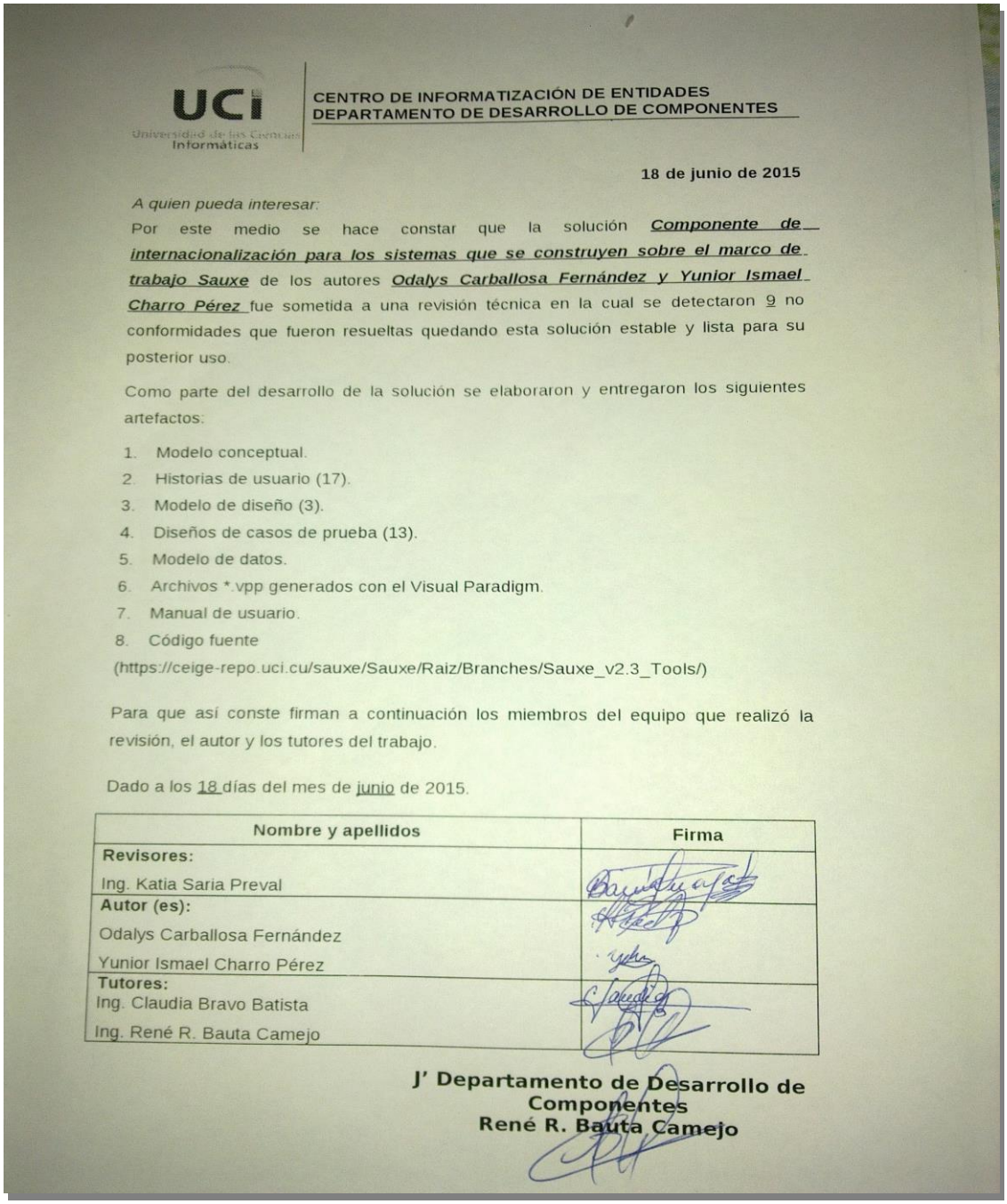
Claudia Bravo Batista
Desarrollador

Yoenry Venega Hechavarría
Desarrollador

René R. Bauta Camejo
Jefe Departamento de Desarrollado de Componentes

Figura 28. Acta que certifica que el componente Internacionalización fue integrado al sistema Xedro - Apside

Anexo 7 Acta de aceptación emitida por el departamento de Desarrollo de Componentes



UCI
Universidad de las Ciencias Informáticas

CENTRO DE INFORMATIZACIÓN DE ENTIDADES
DEPARTAMENTO DE DESARROLLO DE COMPONENTES

18 de junio de 2015

A quien pueda interesar:

Por este medio se hace constar que la solución ***Componente de internacionalización para los sistemas que se construyen sobre el marco de trabajo Sauxe*** de los autores ***Odalys Carballosa Fernández y Yunior Ismael Charro Pérez*** fue sometida a una revisión técnica en la cual se detectaron 9 no conformidades que fueron resueltas quedando esta solución estable y lista para su posterior uso.

Como parte del desarrollo de la solución se elaboraron y entregaron los siguientes artefactos:

1. Modelo conceptual.
2. Historias de usuario (17).
3. Modelo de diseño (3).
4. Diseños de casos de prueba (13).
5. Modelo de datos.
6. Archivos *.vpp generados con el Visual Paradigm.
7. Manual de usuario.
8. Código fuente (https://ceige-repo.uci.cu/sauxe/Sauxe/Raiz/Branches/Sauxe_v2.3_Tools/)

Para que así conste firman a continuación los miembros del equipo que realizó la revisión, el autor y los tutores del trabajo.

Dado a los 18 días del mes de junio de 2015.

Nombre y apellidos	Firma
Revisores: Ing. Katia Saria Preval	
Autor (es): Odalys Carballosa Fernández Yunior Ismael Charro Pérez	
Tutores: Ing. Claudia Bravo Batista Ing. René R. Bauta Camejo	

J' Departamento de Desarrollo de Componentes
René R. Bauta Camejo

Figura 29. Acta de aceptación emitida por el departamento de Desarrollo de Componentes

Anexo 8 Resultado de la prueba de rendimiento utilizando la herramienta JMeter

Informe Agregado

Nombre: Informe Agregado

Comentarios

Escribir todos los datos a Archivo

Nombre de archivo Navegar... **Log/Mostrar sólo:** Escribir en Log Sólo Errores Éxitos **Configurar**

Etiqueta	# Muestras	Media	Mediana	Linea de 90%	Mín	Máx	% Error	Rendimiento	Kb/sec
/seguridad/i...	1	429	429	429	429	429	0,00%	2,3/sec	23,9
/seguridad/...	1	17	17	17	17	17	0,00%	58,8/sec	1645,8
/seguridad/...	1	13	13	13	13	13	0,00%	76,9/sec	538,5
/seguridad/...	1	9	9	9	9	9	0,00%	111,1/sec	423,2
/seguridad/...	1	3	3	3	3	3	0,00%	333,3/sec	1957,7
/seguridad/...	1	2	2	2	2	2	0,00%	500,0/sec	1917,0
/lib/idiomas/...	1	20	20	20	20	20	0,00%	50,0/sec	514,4
/seguridad/i...	1	193	193	193	193	193	0,00%	5,2/sec	55,3
/seguridad/i...	1	194	194	194	194	194	0,00%	5,2/sec	55,2
/seguridad/i...	1	202	202	202	202	202	0,00%	5,0/sec	53,1
/seguridad/i...	1	194	194	194	194	194	0,00%	5,2/sec	55,1
/seguridad/i...	1	173	173	173	173	173	0,00%	5,8/sec	61,7
/seguridad/i...	1	197	197	197	197	197	0,00%	5,1/sec	54,4
/seguridad/i...	1	220	220	220	220	220	0,00%	4,5/sec	48,5
/seguridad/i...	1	215	215	215	215	215	0,00%	4,7/sec	49,6
/seguridad/i...	1	195	195	195	195	195	0,00%	5,1/sec	54,8
/seguridad/i...	1	213	213	213	213	213	0,00%	4,7/sec	50,2
/seguridad/i...	1	166	166	166	166	166	0,00%	6,0/sec	64,6
Total	18	147	193	215	2	429	0,00%	6,6/sec	68,8

Figura 30. Resultado de la prueba de rendimiento utilizando la herramienta JMeter

GLOSARIO DE TÉRMINOS

A.

Artefactos: productos tangibles del proyecto que son creados, modificados y usados dentro de las actividades. Pueden ser modelos, elementos dentro del modelo, código fuente y ejecutables.

C.

Componente: unidad de composición de aplicaciones de software, que posee un conjunto de interfaces y un conjunto de requisitos, y que ha de poder ser desarrollado, adquirido, incorporado al sistema y compuesto con otros componentes de forma independiente, en tiempo y espacio.

Calendario: es una cuenta sistematizada del transcurso del tiempo, utilizado para la organización cronológica de actividades. Se trata de un conjunto de reglas o normas que tratan de hacer coincidir el año civil con el año trópico.

D.

Diagrama: es un gráfico que representa un proceso o refleja relaciones entre datos números que han sido tabulados previamente.

E.

Entidad: forma parte de la estructura de un país y puede comportarse como un ministerio, empresa, asociación, área o cargo.

H.

Herramienta: es un objeto elaborado a fin de facilitar la realización de una tarea. Sirve como recurso.

M.

Métrica: medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado.

S.

Sistema: es un conjunto organizado de objetos o partes interactuantes e interdependientes, que se relacionan formando un todo unitario y complejo.

Software: conjunto de instrucciones que los ordenadores emplean para manipular datos.